AUTOMATICĂ • MANAGEMENT • CALCULATOARE

amc

G. Färber (R.F.G.), R. Radner (S.U.A.), V. N. Novoselțev (U.R.S.S.), P. Kovanic (R.S.C.), J. Demetrovics (R.P.U.), I. Horváth-Gaudi (R.P.U.), G. Buvár (R.P.U.), H. Yoshikawa (Japonia), V. A. Viktorov (U.R.S.S.), V. I. Șumakov (U.R.S.S.), A. Békéssy (R.P.U.), L. Hannák (R.P.U.), Cs. Balogh (R.P.U.)	„O PUNTE ÎNTRE ȘTIINȚĂ ȘI TEHNOLOGIE" (Congresul Mondial Trienal al Federației Internaționale de Automatizare IFAC – Budapesta '84) • Conducere cu calculator a proceselor • Sistem de fabricație flexibilă în Japonia • Jocuri de întreprindere în SUA • Reglare cu calculatoare în industrie, medicină, biologie, agricultură
G. Gromov (U.R.S.S.)	RESURSELE INFORMAȚIONALE NAȚIONALE (Ciclul „Display")
T. Geber, E. Miclescu	MINICALCULATOARELE CORAL ȘI INDEPENDENT (II) (Ciclul „Service pentru calculatoare")
G. Manolescu, A. Varga, A. Davidoviciu	PROIECTAREA ASISTATĂ DE CALCULATOR (Consfătuire națională CPAC '84)
A. Petrescu, N. Țăpuș, Fl. Tănase, T. Domocoș, G. Drăghicescu, A. Negulescu, Fr. Iacob, T. Moisa, I. Atanasiu, D. Mihu, A. Anghel, C. Alupului	MICROCALCULATOARELE: PERSONAL–PROFESIONAL FELIX PC, PERSONAL HC-80 (Ciclul „Calculatoare personale")
A. Borozan, R. Ciocea, M. Ivan, Șt. Panait, O. Borozan, M. Constantinescu, A. Lazăr, D. Trifănescu	MEMENTO DE TELEPRELUCRARE (II) (Ciclul „Rețele de calculatoare")
I.I.A.S.A.*	„FERIȚI-VĂ DE CAPCANE ÎN ANALIZA DE SISTEM" (Ciclul „Analiza sistemelor")
M. Drăghici	PENTRU BAZE DE DATE RELAȚIONALE! (Ciclul „Opinii interactive")

49

SERIE CONTINUĂ DE SINTEZE, CERCETĂRI APLICATIVE, INSTRUIRE, INFORMARE ÎN SISTEME AUTOMATE, INFORMATICE, ELECTRONICE DE CONDUCERE

amc

Vol. 49

AUTOMATICĂ
MANAGEMENT
CALCULATOARE

- Ciclul special „Congresul mondial IFAC '84"
- Resursele informaționale naționale
- Minicalculatoarele CORAL și INDEPENDENT
- PAC
- Microcalculatoare personale românești
- Memento de teleprelucrare
- Analiză sistem



EDITURA TEHNICĂ
BUCUREȘTI – 1985

a
m
c
Vol. 49

Traduceri, adaptări: Adrian Davidoviciu, Vasile Sima, Radu Magda, Marcel Sîrbu, Aristide Predoi, Sandu Lazăr, Dan Dobrescu, Alexandra Tatu, Matei Constantin, Doina Tudor, Nicolae Tudorancescu, Ilie Păiuşi, Cornelia Basarab, Dan Rădulescu, Virgil Micula, Şerban Sporea, Radu Manolescu, Nicolae Algiu, Leonard Stoian Horobeţ, Gabriel Spiridon, Eust. Stanciu, Ştefan Lupescu.

Recenzii: Adrian Davidoviciu, Geber Toma, Gheorghe Luchian, Gorun Manolescu Sandu Lazăr, Liviu Dumitraşcu

Redactor: ing. **PAUL ZAMFIRESCU**

Tehnoredactor: **Maria Trăznea**
Coperta: **Simona Dumitrescu**

Bun de tipar: 12 sept. 1985.
Coli de tipar: 21.

Tiparul executat la Întreprinderea Poligrafică „Crişana", Oradea — cda. nr. 32/1985.



amc
Vol. 49

CUPRINS

În paginile 43, 116 sînt cuprinse manifestări naționale și internaționale, recomandările de cărți ș.a., astfel: recomandări seria AMC 1985.

P.3. CONDUCEREA PROCESELOR TEHNOLOGICE ŞI SISTEMELE INFORMATICE

G. Färber

Catedra de calculatoare de proces,
Universitatea Tehnică din München,
R.F.G.

REZUMAT. Sistemele de conducere cu calculator a proceselor tehnologice nu mai reprezintă ca pînă acum insule izolate de prelucrare automată a informaţiilor. Ele au nevoie de o strînsă interconectare cu sistemul informatic al întregii întreprinderi. Informaţii actualizate privind modul de desfăşurare a proceselor de producţie trebuie transmise sistemului de conducere, iar datele de plan şi alte informaţii prioritare trebuie transmise în celălalt sens. Lucrarea de faţă discută realizările prezente şi unele tendinţe în realizarea sistemelor de conducere cu calculatoare a proceselor tehnologice, analizînd posibilităţile tehnice de interconectare cu sistemele informatice de conducere a întreprinderilor.

Introducere

Utilizarea în mediu industrial a tehnicii de calcul electronic cuprinde atît conducerea proceselor tehnologice cît şi gestiunea economică a întreprinderilor. Ele au fost dezvoltate, însă, în medii foarte diferite din punct de vedere organizatoric:

— Colectivele tehnice care răspundeau pînă acum de aparatura convenţională de automatizare răspund acum de proiectarea şi implementarea calculatoarelor de conducere a proceselor tehnologice. Ele sînt în continuare subordonate conducerii ce răspunde de producţie în cadrul întreprinderilor.

— Colectivele de informatică au început cu lucrări mai simple de gestiune („era cartelei perforate") şi în timp au preluat tot mai multe responsabilităţi. Actualmente, cînd se realizează sisteme informatice integrate de conducere, aceste colective dobîndesc o mare influenţă asupra conducerii unităţilor.

Actualmente, cele două tipuri de aplicaţii ale tehnicii de calcul electronic se dezvoltă împreună şi se suprapun parţial: pe de o parte ele trebuie să facă schimb reciproc de date, iar pe de altă parte este o anumită concurenţă în dezvoltarea de noi aplicaţii şi funcţii.

Sistemul integrat de conducere, care se extinde asupra tuturor activităţilor, poate avea ca origine sistemul de conducere cu calculator a proceselor tehnologice, în care caz este bazat, de regulă, pe echipamente de calcul de capacitate medie. El poate fi, însă, şi o extindere a sistemului informatic al întreprinderii, respectiv un nod puternic într-un sistem distribuit de prelucrare a datelor. Oricum, există o interfaţă între

sistemul informatic de gestiune și cel de conducere a proceselor tehnologice, care cuplează între ele cele două sfere de conducere.

O asemenea interconectare nu este ușor de realizat, ea generează probleme tehnice, cum și probleme de responsabilități împărțite între grupuri diferite care au fost independente pînă acum.

Lucrarea urmărește să ofere o imagine asupra nivelului atins în realizarea sistemelor de conducere cu calculator a proceselor tehnologice temele informatice de conducere. Sînt discutate și unele dificultăți în realizarea acestor interconectări precum și necesitățile unor standarde de comunicație.

Sisteme de conducere cu calculator a proceselor tehnologice

Funcțiuni și echipamente

Pentru aplicații de conducere cu calculator a proceselor tehnologice este relativ ușor să identificăm clase de funcțiuni ce trebuie realizate de echipamente bazate pe tehnică de calcul electronic. Iată cîteva exemple:

— Culegerea de date și controlul centralizat (înregistrare date)
— Funcții de supraveghere
— Comenzi secvențiale (ca la automatele programabile)
— Reglare numerică (sisteme în buclă închisă)
— Comunicare om-mașină (inclusiv pentru afișarea rezultatelor și interactivitate)
— Gestiunea datelor (cum ar fi bazele de date).

Multe din aceste funcțiuni pot fi standardizate și parametrizate; suplimentar, de regulă, există și alte prelucrări dependente de aplicația concretă.

Aceste funcțiuni pot fi implementate cu ajutorul unor echipamente universale, cum este cazul sistemelor convenționale de conducere cu calculator a proceselor tehnologice. Același tip de echipament (de regulă bazat pe microprocesoare) este folosit pentru realizarea diferitelor funcțiuni care sînt implementate prin software. O asemenea abordare duce implicit la costuri ridicate pentru implementările individuale.

Se observă, totodată, o tendință către realizarea de echipamente specializate. Pentru a reduce complexitatea sistemului, la asemenea echipamente, un anumit dispozitiv efectuează o singură funcție. Deseori se folosesc arhitecturi de echipamente dedicate și optimizate, susținute de sisteme de elaborare și punere la punct a programelor aplicative aferente acestora. Exemple de asemenea echipamente sînt automatele programabile pentru comanda secvențială, echipamentele specializate de control centralizat sau sistemele distribuite de reglare automată (de exemplu, TDC 2000).

Aceste echipamente specializate sînt foarte utile în aplicațiile de mică complexitate. Foarte frecvent toate aceste funcțiuni menționate mai

sus există simultan într-o anumită combinație, astfel încît dispozitivele disparate specializate trebuie — cel puțin — să comunice între ele. Din păcate, posibilitățile lor de interconectare sînt foarte puțin dezvoltate.

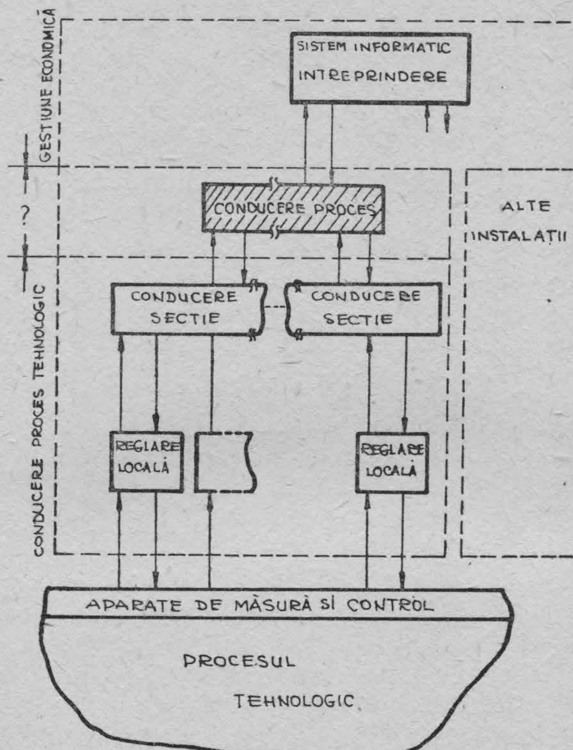


Fig. 1. Structura ierarhizată a sistemului de conducere a producției.

Integrarea în sisteme

Este o problemă destul de dificilă de a integra echipamentele izolate într-un sistem unic. Rețeaua care rezultă este capabilă de a asigura schimbul de informații și permite utilizarea în comun a instrumentației (de exemplu, a traductoarelor), între diferitele funcțiuni.

Este evidentă necesitatea pentru o rețea de interconectare care să asigure transferul fizic al informațiilor. Aplicațiile de conducere cu calculator a proceselor tehnologice au tendințele de a utiliza sisteme cu magistrale. În fig. 2 se ilustrează faptul că sistemele distribuite actuale nu se bazează doar pe un singur tip de magistrală: în schimb, există o ierarhie de magistrale ce interconectează toate dispozitivele inteligente, programabile (Wiemann și colaboratorii, 1981—1984):

— s-au depus eforturi considerabile pentru standardizare în domeniul magistralelor de date pentru aplicații de conducere a proceselor teh-

nologice (PROWAY-Walze, — 1978), care interconectează calculatoare frontale la sisteme mai puternice de conducere automată a proceselor tehnologice;

— o magistrală de date la nivelul întreprinderii, cu o bandă de transmisie mai largă poate lega între ele aceste sisteme de conducere a proceselor tehnologice. În viitorul apropiat se vor asigura viteze de trans-

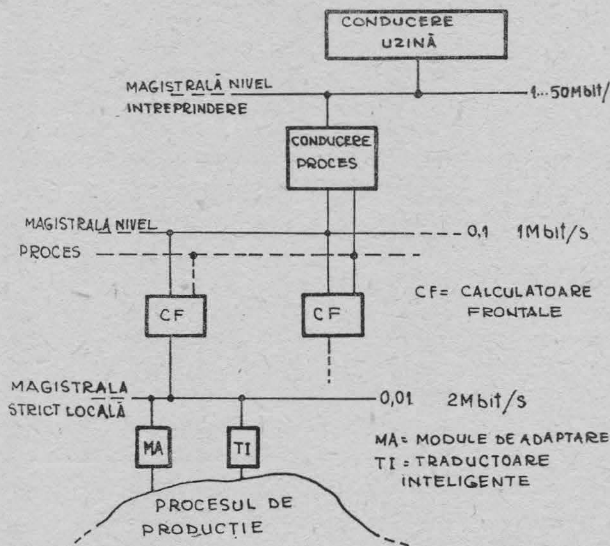


Fig. 2. Ierarhia de magistrale de sistem.

misie de pînă la 100 Mbiți/s; actualmente sistemul ETHERNET, cu o viteză de transmisie de 10 Mbiți/s, poate constitui o soluție bună (IEEE, 1981);

— la nivelul cel mai de jos, o magistrală strict locală va înlocui cablajul actual foarte costisitor dintre sistemul de automatizare și instrumentația din procesul tehnologic (Färber, 1983). Vitezele de transmisie pot varia între 10 kbiți/s și 2 Mbiți/s, valori ce sînt tehnologic realizabile (MacWilliams, 1984) și permit astfel de viteze foarte mari de transmisie între traductoare și calculatorul frontal.

În afara acestor magistrale seriale de sistem, toate echipamentele au magistralele lor interne, paralele, care interconectează modulele locale.

Mai importantă decît interconectarea fizică este posibilitatea de a face schimb de date care să fie corect interpretate la adresantul receptor. În această problemă este larg acceptat modelul OSI cu 7 niveluri (OSI = = Open Systems Interconnection sau Interconectarea Sistemelor Deschise). Nivelurile de jos, pînă la nivelul 4 inclusiv, sînt obiectul unor standarde sau propuneri de standardizare, dar — din nefericire — nu există încă propuneri, comun agreate, pentru rețelele specifice sistemelor informatice de conducere la nivel de întreprinderi și pentru rețelele

specifice conducerii proceselor tehnologice. Și mai dificilă este definirea nivelelor 5 și 6:

— nivelul de sesiune (5) stabilește un limbaj comun de comandă, înțeles de diferitele sisteme de operare;

— nivelul de prezentare (6) răspunde de reprezentarea internă a informației structurate (de exemplu, o matrice a valorilor provenite de la traductoare sau un text) cum și de prezentarea datelor pentru uzul programelor aplicative.

Astăzi, programatorul de aplicații răspunde de interpretarea tuturor biților transmiși, trebuie să se îngrijească de comenzile spre sistemele de operare și de structura datelor. În viitor, un standard comun va ajuta programatorul să folosească datele și comenzile pentru alte sisteme în același mod ca pentru sistemul de calcul la care este conectat.

Sistemele de reglare automată distribuite disponibile astăzi oferă capabilități încă limitate: ele se pot aplica conform scopului propus dar sînt dificil de programat, de exemplu, în cazul algoritmilor de comandă secvențială pentru echipamentul TDC 2000. Sistemul Teleperm M (Siemens (Janetzky și colaboratorii, 1983)) oferă o mai mare flexibilitate, iar unele noi sisteme anunțate (de exemplu, sistemul TDC 3000 al firmei Honeywell) par să satisfacă toate cerințele unui sistem integrat — cu dezavantajul important că la sistem se pot lega doar aparataje de la un anumit furnizor.

Programarea și distribuția funcțională

Programarea calculatoarelor convenționale de conducere a proceselor tehnologice are loc în majoritatea cazurilor într-un mediu cunoscut de lucru:

— se dispune de un sistem de operare în timp real, care permite multiprogramarea și asigură timpi reduși de răspuns la stimulii externi;

— programele sînt scrise în limbaje universale cum sînt FORTRAN sau PASCAL; limbajele sînt de regulă extinse pentru a permite multiprogramarea și operații de intrare/ieșire de/la interfețele de proces. Unele limbaje mai noi de programare au standardizat asemenea facilități cum sînt limbajele PEARL (DIN 66253, 1981, 1982); ADA și ca o variantă mult mai puțin complexă și mai elegantă — MODULA-2, care asigură mecanismele de bază pentru multitasking, suportînd totodată funcții de nivel inferior. Nu se remarcă însă progrese semnificative în utilizarea lor, multe aplicații continuînd să fie programate în maniera bine cunoscută de programare algoritmică detaliată.

Același mediu de programare este utilizat și pentru sistemele cu mai multe calculatoare. În fig. 3 se ilustrează un exemplu asupra modului în care apelurile unui sistem de operare local (de exemplu, pentru comunicările între procese) pot fi extinse spre un serviciu global: la un nivel APD (Apel la o Procedură de la Distanță) fiecare apel la sistem este identificat a fi:

— local: în care caz este îndrumat spre sistemul local de operare și este executat local;

— sau la distanță: folosind Software-ul de Comunicație (SC) și Echipamentul de Comunicație (EC), apelul este îndrumat spre un sistem de operare de la distanță iar răspunsul este transmis înapoi și îndrumat spre apelantul local.

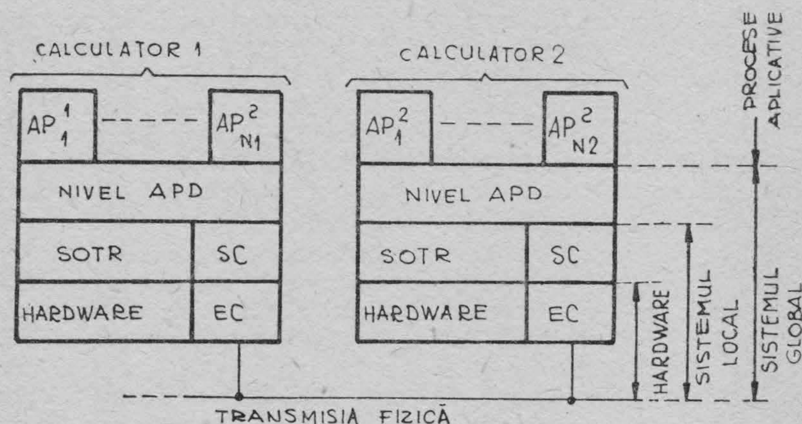


Fig. 3. Arhitectura sistemului distribuit.

Pentru un programator, execuția la distanță fiindu-i transparentă, sistemul se comportă ca un singur calculator (transparență).

Sistemele distribuite de conducere cu calculator a proceselor tehnologice utilizează, de asemenea, pachete de programe aplicative adecvate, care permit utilizarea unor module funcționale generalizabile (cum ar fi cele corespunzătoare unor algoritmi de comandă, culegerea de date de la traductoare etc.). Aceste module pot fi reutilizate în multe aplicații și astfel tind să devină din ce în ce mai sigure în funcționare. Pentru programator ele apar sub forma unui limbaj de tip schemă-bloc cu ajutorul căruia modulele funcționale pot fi apelate ca niște blocuri ca într-un mediu grafic (Korn, 1977): blocurile sînt generate, interconectate și parametrizate; sistemul generează în mod automat un program executabil. Se pot elabora noi module într-un limbaj convențional de programare și care se pot adăuga bibliotecii de module. În timp ce primele generații a acestor sisteme de programare erau limitate la o singură clasă de aplicații (de exemplu, comanda secvențială), noile sisteme de programe permit — de asemenea — combinarea diferitelor funcțiuni.

În fig. 4 se indică modul în care într-un sistem multicalculator, diferitele funcțiuni pot fi distribuite între diferitele calculatoare individuale. În cazul sistemelor mono-calculator, toate aceste funcțiuni sînt preluate de unul și același calculator, în timp ce,

— distribuirea *orizontală* specializează calculatoarele, pentru a executa o singură clasă de funcțiuni (de exemplu, sisteme de supraveghere, sisteme pentru reglare numerică directă etc.) iar,

— distribuirea *verticală* asigură preluarea de către toate calculatoarele a tuturor funcțiilor dar asigură împărțirea încărcării între atîtea calculatoare cîte se dovedesc necesare.

Prima abordare este mai ușor de însușit și de implementat, iar echipamentele pot fi specializate pentru diferitele activități. Distribuirea verticală permite însă o mult mai bună adaptare la performanțele dorite

FUNCTIA	SISTEM CENTRALIZAT	SISTEM ORIZONTAL	DISTRIBUIT VERTICAL
CULEGERE DATE		C1	
SUPRAVEGHERE		C2	C1 · CM
COMANDĂ			
OM-MASINĂ		CM	

Fig. 4. Distribuirea orizontală și verticală a funcțiilor între sisteme multi-calculator.

și la structura procesului fizic comandat. Subprocesele (ce necesită funcțiuni foarte diferite) pot fi asociate unor calculatoare anume, toate prelucrările relative unui subproces avînd loc în unul și același sistem, transmiterea la distanță a datelor fiind minimizată. Este ușor de obținut și o fiabilitate corespunzătoare: o defecțiune, de exemplu, într-un sistem de supraveghere a procesului tehnologic poate compromite orice altă funcție de comandă secvențială asupra procesului, în timp ce o defecțiune a unui calculator dedicat unui subproces va afecta doar acel subproces. Este de asemenea mult mai ușor a asigura o anumită redundanță atunci cînd componentele ce trebuiesc dublate sînt toate identice.

Tendințe viitoare

Principala motivație pentru noi dezvoltări a sistemelor de reglare automată a proceselor tehnologice este de regulă reducerea costurilor. Unele aspecte importante ale viitoarelor sisteme sînt următoarele:

— reducerea costurilor pentru cablaje și instalare care pot reprezenta pînă la 50% din costul total al sistemelor;

— reducerea costurilor de elaborare a software-ului prin utilizarea unor instrumente perfecționate de programare și medii-suport eficiente de programare;

— reducerea costurilor cauzate de defecțiunile sistemului: adăugarea unei redundanțe (echipamentele devin din ce în ce mai ieftine) reduce probabilitatea defectării cu un ordin de mărime și economisește costurile ce ar rezulta din producerea unor defecte.

Pentru a reduce costurile de instalare, tehnologia semiconductorilor permite înlocuirea cablurilor scumpe prin module funcționale inteligente.

În fig. 5 se exemplifică o „magistrală strict locală“ ce transmite datele și tensiunea de alimentare a modulelor de adaptare care asigură funcțiile locale de intrare/ieșire dinspre/spre proces. Acestea pot fi instalate imediat lîngă proces iar întregul cablaj se reduce la o magistrală formată din 2 fire. Nucleul modulului de adaptare este un microcalculator pe o singură pastilă care realizează funcțiile de comunicație și de intrare/ieșire

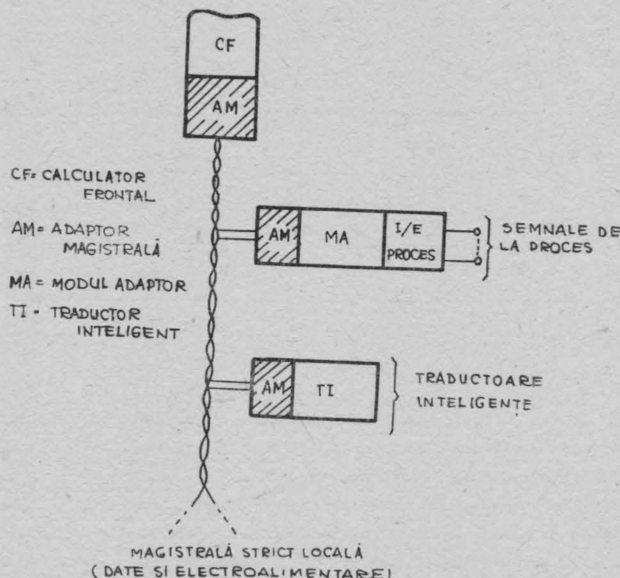


Fig. 5. Sisteme de magistrale strict locale.

de la/la proces, inclusiv toate funcțiunile locale de supraveghere și diagnosticare. Există microcircuite pre-programate (de exemplu, MOSTEK SCU 20 (Mostek, 1982)) cu viteze de transfer de pînă la 9 600 biți/s. Noile produse anunțate recent (Intel 8044 (MacWilliams, 1984)) indică faptul că restricțiile de viteză vor dispare curînd: se vor realiza viteze de pînă la 2,4 Mbiți/s. Pe un orizont mai lung, se pare că aceste module vor fi integrate în traductoare („traductoare inteligente“) și în elemente de execuție, astfel încît cablajul este redus la interconectarea acestor dispozitive printr-un cablu bifilar. Standardizarea unei „magistrale strict locale“ devine o necesitate.

O scurtă notă este necesară pentru calculatoarele personale profesionale care devin din ce în ce mai ieftine și care sînt și ele utilizate în aplicații de control centralizat. Stabilitatea lor pe piață (producție de mare serie) le face potrivite de asemenea pentru aplicații, în general de mici proporții, de conducere a proceselor tehnologice. Există deja pe piață module de I/E de proces, compatibile între ele din punctul de vedere al magistralei de interconectare, precum și claviaturi și display-uri (cu facilități grafice și color) ce asigură o bună interfață între om și sistem.

O cerință pentru a face față faimoasei „crize a software-ului" este existența unui mediu de programare. Un sistem de calcul care oferă de regulă un asemenea mediu, în mod normal nu asigură cerințele de lucru în timp real a sistemului de conducere cu calculator a proceselor tehnologice. Fig. 6 indică o configurație de sistem cuprinzînd 2 tipuri de sisteme interconectate care pot rezolva acest conflict (Ritchie, 1978):

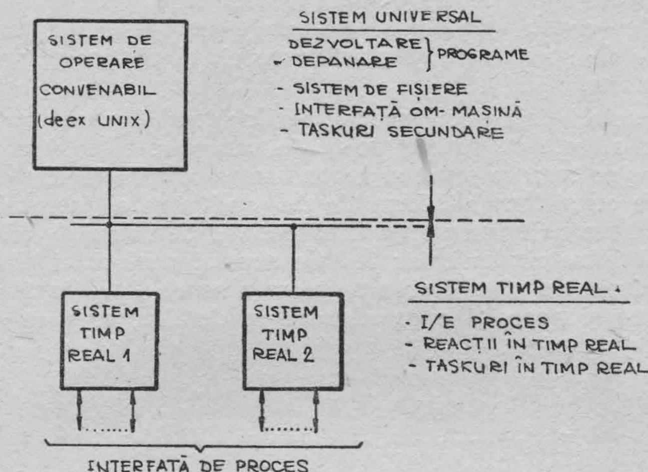


Fig. 6. Sistem universal și sisteme în timp real.

— Un sistem universal bazat pe un sistem de operare convenabil (cum ar fi UNIX) ce este folosit pentru elaborarea, inclusiv depanarea, programelor de timp real. Aceste programe sînt legate între ele și încărcate în sistemele cu funcționare în timp real unde sînt executate în condițiile restrictive de timp real. Sistemul universal efectuează de asemenea funcțiunile „lente" cum sînt accesurile la fișiere, interacțiunea om-sistem sau efectuarea unor prelucrări în partiții secundare.

— Una sau mai multe sisteme în timp real bazate pe un sistem de operare în timp real eficient, care efectuează operațiunile efective de intrare/ieșire cu procesul precum și toate celelalte operații critice ca timp. Aceste programe au fost încărcate în aceste calculatoare în timp real de pe sistemul de calcul universal, pe legătura ce le leagă; cînd este necesar un acces la sistemul de fișiere sau o intervenție a operatorului tehnolog, se activează taskul (lent) corespunzător din calculatorul universal.

Sistemul de operare convențional monolitic de conducere a procesului tehnologic dispare, fiind înlocuit printr-un sistem standard pe unul din calculatoare (identic cu sistemele folosite în aplicațiile universale) și cu sisteme de tip executiv de timp real specializate, de volum mic, pe celelalte calculatoare. Combinația între aceste sisteme oferă un mediu omogen și facil pentru programatorii de aplicații.

Noi metode de elaborare a programelor pătrund și în domeniul programării aplicațiilor de conducere a proceselor tehnologice. S-au adoptat

metode de inginerie a programării (de exemplu, sistemul EPOS (Biewald și colab., 1979)) precum și tehnici noi de programare specifice aplicațiilor de conducere a proceselor tehnologice. Utilizarea de baze de date optimizate pentru cerințele de timp real simplifică implementarea multor aplicații; programarea funcțională sau logică (de exemplu, limbajul PROLOG (Clocksin, Mellish, 1981)) eliberează programatorul de sarcina de a detalia modul în care o anumită problemă trebuie rezolvată: el trebuie doar să specifice în ce constă problema. Sistemele expert vor ajuta și ele atît pentru fazele de specificare și planificare cît și pentru partea de comunicare om-sistem.

În fine, sistemele de conducere cu calculator a proceselor tehnologice trebuiesc completate cu funcții de ordin superior. Este vorba de funcții de planificare pe termen scurt și respectiv pe termen mediu care necesită o comunicare cu sistemul informatic de conducere al întreprinderii; planificarea pe termen lung se execută de obicei pe calculatorul de gestiune economică al întreprinderii, iar ieșirile acestei prelucrări trebuiesc folosite ca intrări pentru planificarea pe termen scurt. Funcțiunile se pot parțial suprapune și necesită o coordonare în detaliu, ceea ce constituie și o problemă de strategie.

Integrarea în sisteme informatice

Arhitectura sistemelor informatice

Arhitectura echipamentelor și a programelor pentru sistemele informatice orientate pe aplicații de gestiune economică este oarecum dominată de filozofia firmei IBM. Un număr de alte tipuri de arhitecturi de sisteme originale au fost înlocuite de cele de tip IBM 370 și succesorii săi moderni (4 300, 308×). Doar un număr mic de producători de unități centrale de calcul compatibile cu IBM 370 (Fujitsu, Amdahl) au reușit să obțină o parte mai semnificativă din piață.

Același fenomen s-a produs și pentru sistemele de operare: Sistemele de operare VM și MVS ale lui IBM reprezintă baza oricărei investiții serioase în dezvoltarea de software de către utilizator. În multe firme, direcția promovată de IBM în domeniul programelor este un lucru stabilit, ce se urmărește fără devieri.

Sistemele informatice au și ele tendința de a deveni distribuite. Nodurile inteligente migrează spre instalația tehnologică și preiau o parte a sarcinii de prelucrare de la unitățile centrale mari; structura de bază este foarte similară cu cea de la sistemele distribuite de conducere cu calculator a proceselor tehnologice. Din nou, firma IBM este aceea care dictează standardul pentru interconectarea nodurilor și terminalelor; SNA (arhitectura de sistem de rețea) comandă transmiterea logică a informațiilor și orice alt sistem ce necesită acces la sistemul informatic trebuie să respecte un asemenea standard (Randesi, 1982).

Sistemele informatice au nevoie, de asemenea, de a avea acces la sistemele de conducere a proceselor tehnologice pentru a asigura o creștere a eficienței activității la nivelul întregii întreprinderi:

— sînt necesare date reale referitoare la modul de desfășurare a producției, care să permită perfecționarea procesului de planificare și de control al calității sau de depistare a punctelor de gîtuire din procesul de producție;

— este de asemenea important de a transmite prompt informații noi sistemului de conducere a procesului tehnologic, de exemplu informații referitoare la planificarea pe termen mai lung pentru a fi folosite local sau informații de comandă care să influențeze prioritățile de fabricație funcție de cererile beneficiarilor.

Cerința pentru interconexiuni cît mai strînse între sistemele informatice și cele de conducere a proceselor tehnologice crește pe măsură ce producția devine din ce în ce mai automatizată și trebuie să se adapteze cerințelor dinamice de piață cît mai prompt și flexibil.

Firmele mai mici folosesc uneori echipamente non-IBM; de regulă, sistemele lor informatice se bazează pe calculatoare de capacitate medie, de exemplu pe cele produse de firma DEC (PDP-11, VAX). Comunicarea cu sistemele de conducere a proceselor tehnologice este realizată pe conceptul arhitecturii de rețea promovată de furnizor (de exemplu, DECNET). Din păcate rețelele diferiților furnizori nu sînt direct compatibile între ele. Din această cauză un efort apreciabil de adaptare este preluat de firme specializate pe integrarea de sisteme și pe elaborarea de software de comunicații.

Probleme de interconectare

Interconectarea rețelelor cu arhitecturi diferite se poate realiza pe o cale mai mult sau mai puțin generală: de la o linie de interconectare asincronă serială simplă exploatată prin programe de aplicații speciale în cele 2 capete ale liniei, pînă la o integrare completă a tuturor funcțiilor rețelei.

Una din cele mai frecvente soluții care permite transmiterea reciprocă de date între două sisteme este emularea într-o rețea a unui terminal-standard (de exemplu de tip IBM 3780, 3270) printr-un program care se execută într-unul din nodurile celeilalte rețele. Fig. 7 indică structura hardware/software care trebuie implementată:

— în partea în care se emulează (în cazul de față sistemul de conducere a procesului tehnologic) trebuie asigurate echipamentul de comunicație și software-ul de protocol (de exemplu, de tip BSC, SDLC); după aceasta cele două sisteme își pot transmite reciproc date sub formă de secvențe de baiți;

— în ambele părți trebuiesc elaborate programe aplicative suplimentare care să fie capabile să interpreteze datele într-un mod consistent, aceste programe fiind dependente de aplicația concretă.

Metoda emulării de terminale oferă însă posibilități limitate și necesită un volum mare de elaborări individuale particulare. Ca o soluție și mai puțin costisitoare, se poate utiliza un convertor de protocol care să

efectueze toate funcţiile complexe de conversie printr-un bloc hardware suplimentar (Bracker, 1983).

O a doua soluţie foloseşte unele caracteristici ale unor arhitecturi dintr-o reţea (de exemplu, SNA) şi asigură comunicaţii multiple interactive între task-urile din cele două sisteme. În mod concret, se pot folosi programe standard din mediul SNA, fără nici un efort de adaptare (de

Fig. 7. Comunicaţie prin emulare terminal.

exemplu, acces la pachetul de programe de gestiune economică IMS). Adică:

— nodul reprezentat de sistemul de conducere a procesului tehnologic va emula un dispozitiv SNA, de tip PU-2 (PU=physical unit=unitate fizică), care permite la rîndul lui o comunicare multiproces prin multiplexare (mai multe legături de comunicare simultane ce pot folosi acelaşi circuit fizic);

— în nodul reprezentat de sistemul de conducere a procesului tehnologic se pot emula două sau mai multe terminale (de exemplu, de tip IMB 3278); aceasta permite ca orice terminal conectat la sistemul de conducere a procesului să aibă acces la programele de aplicaţie IBM standard;

— se pot stabili de asemenea şi legături logice între task-urile din sistemul de conducere de proces şi mediul SNA.

Metoda descrisă mai sus oferă mai multe facilităţi, dar reflectă o relaţie uni-direcţională: task-urile sistemului de conducere a procesului

pot avea acces la task-urile SNA, dar task-urile SNA nu se potrivesc cu arhitectura de comunicare în rețea a sistemului de conducere a procesului tehnologic.

Pentru a ajunge la o situație complet simetrică, porțile de legătură între cele două sisteme trebuie să fie simetrice: trebuiesc implementate facilitățile de domenii încrucișate. În terminologia SNA, un nod PU-4 este capabil să asigure accesul în ambele direcții: oricare terminal al sistemului IBM poate fi folosit pentru oricare task al sistemului de conducere a procesului tehnologic. Deoarece o simetrie completă este greu de realizat și deoarece majoritatea aplicațiilor nu necesită asemenea posibilități, majoritatea interconectărilor actuale sînt realizate ca emulări PU-2 (CSI, 1983).

Majoritatea problemelor care apar în domeniul transmisiei datelor la distanță nu au caracter pur tehnic. Pe de o parte, firmele producătoare de echipamente de calcul au tendința de a ține secret datele despre protocoalele folosite, parțial din motive de strategie de piață și parțial deoarece este foarte dificil de a furniza specificații detaliate și complete. Pe de altă parte, responsabilitățile legate de realizarea sistemelor informatice și a sistemelor de conducere a proceselor tehnologice sînt plasate în organizații separate: aceasta generează noi probleme ce nu au caracter tehnic datorită competențelor nedefinite precis.

Soluții privind furnizori unici

O cale de a evita toate problemele suplimentare legate de interconectarea celor două tipuri de sisteme este de a achiziționa echipamentele pentru sistemul informatic și pentru sisteme de conducere a proceselor tehnologice de la același furnizor. Există firme competente în ambele domenii; totuși nu este totdeauna atît de ușor să se interconecteze două rețele livrate chiar de același fabricant (de exemplu, SINET și TRANSDATA al firmei Siemens) datorită departamentelor (filialelor) diferite care răspund de cele două arhitecturi de rețele.

Firma DEC oferă o arhitectură de rețea omogenă. Pornind de la aplicații de conducere a proceselor industriale, firma DEC s-a extins cu sistemele sale spre aplicații de conducere la nivelul secțiilor și chiar al întreprinderilor. Sistemul DECNET oferă o soluție unică pentru întreaga gamă de calculatoare, de la calculatorul pe o placă tip FALCON pînă la mega-minicalculatorul VAX 11-780.

De asemenea firma IBM, pe departe liderul comercial care domină piața, are preocupări istorice în domeniul calculatoarelor de proces. După un început lăudabil cu calculatorul de proces IBM 1800, generațiile care au urmat (sistemele /7 și chiar /1) nu au putut continua succesul inițial. Dar mai recent a fost reactivată o inițiativă mai veche ce utilizează un echipament deja existent de mai mulți ani: fig. 8 indică modul în care sistemul ACS (Advanced Control System = Sistem Avansat de Comandă (IBM, 1983)) interconectează calculatoarele universale (/370) cu cele de proces (/1) și sistemul integrat de programe distribuit pe cele două calculatoare; dialogul sistem-operator este asigurat de o consolă de opera-

tor realizată într-o nouă tehnologie. Acestea sînt instrumente eficiente pentru elaborarea programelor aplicative. Pentru moment acest sistem este limitat la conducerea unor procese tehnologice lente, dar implicarea firmei IBM în automatizarea proceselor de fabricaţie cu caracter discret (sisteme robotizate) indică noi dezvoltări în viitor.

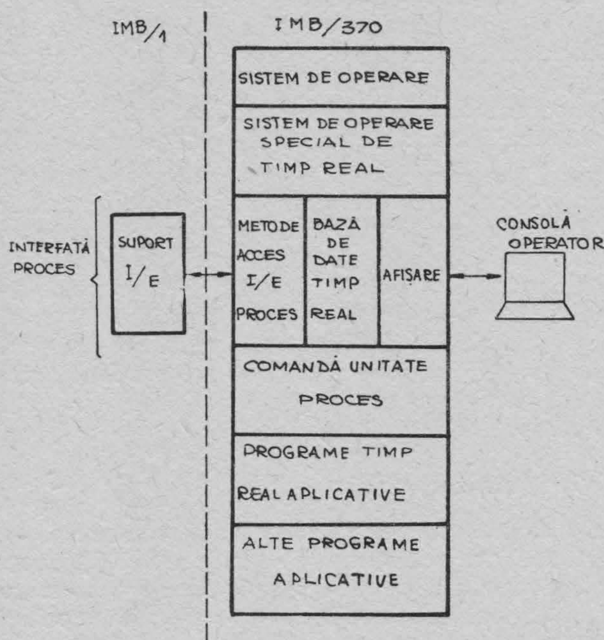


Fig. 8. Structura unui sistem de reglare automată avansată (ACS — Advanced Control System)

Pentru beneficiari, dependenţa strînsă de un singur furnizor poate fi o problemă serioasă: este foarte dificil de extins sistemele existente cu furnituri de la alţi producători care poate sînt mai experţi în anumite domenii sau oferă soluţii mai bune pentru anumite probleme.

Standarde şi interconectarea reţelelor

În subcapitolul 2 sînt tratate diferitele soluţii particulare pentru problemele de interconectare iar în subcapitolul 3 se prezintă soluţia furnizorului unic, ambele soluţii, prezentînd dezavantaje serioase. Soluţia ideală ar fi existenţa unui standard care să fie folosit de toţi producătorii atît de echipamente pentru sistemele informatice cît şi pentru conducerea proceselor tehnologice.

Modelul de interconectare a sistemelor deschise conform propunerii ISO ar putea constitui o bună bază pentru un standard de comunicaţii, dar din păcate cele 7 niveluri nu sînt încă complet specificate. Din contră, apar din ce în ce mai multe propuneri contradictorii şi vor trece pro-

babil încă mulţi ani pînă la convenirea unui standard. Aceasta este de dorit în special pentru nivelurile orientate spre aplicaţii, căci ar permite ca fiecare sistem să înţeleagă toate comenzile de nivel înalt şi reprezentările complexe a datelor din celălalt sistem. Necesităţile pentru sisteme de comunicaţii publice vor accelera procesul de standardizare, dar primele definiri vor fi probabil dedicate prelucrării de texte şi nu aplicaţiilor de conducere a proceselor industriale.

Unele firme oferă reţele ca un serviciu distinct: ele oferă soluţii de echipamente şi software pentru interconectări în reţele chiar a unor calculatoare de la furnizori diferiţi şi cu arhitecturi diferite. De exemplu, reţeaua FUSION (NRC, 1983) oferă un set limitat de funcţii de reţea folosind un protocol intern al firmei (variante ale protocolului XNS sau TCP/IP) şi asigurînd transferul de fişiere şi lucru de la distanţă pentru calculatoare cum ar fi PDP11/RSX, IBM/PC, VAX/VMS şi diverse sisteme sub sistemul de operare UNIX. Mulţi alţi furnizori oferă servicii şi mai simple de transport al informaţiei.

O cale intermediară ar fi utilizarea de porţi de interconectare: similar cu convertoarele de protocol (care realizează aceeaşi funcţie la un nivel inferior al interconectării), acestea adaptează arhitecturile reţelelor prin implementarea a două pachete de programe într-un singur bloc hardware-software. Fig. 9 indică modul în care o parte a unei asemenea porţi de interconectare se conformă tuturor standardelor de protocol ale reţelei cu arhitectura A, informaţiile relevante fiind inter-transmise la nivelul de aplicaţie a părţii, iar cealaltă parte execută protocoalele reţelei cu arhitectura B. Costurile hardware pentru aceste funcţiuni nu trebuie să fie prea mari (de exemplu, există asemenea porţi complete de interconectare realizate pe o singură plachetă în standard MULTIBUS (XICOM, 1983)) dar elaborarea software-ului poate necesita mai mulţi ani.

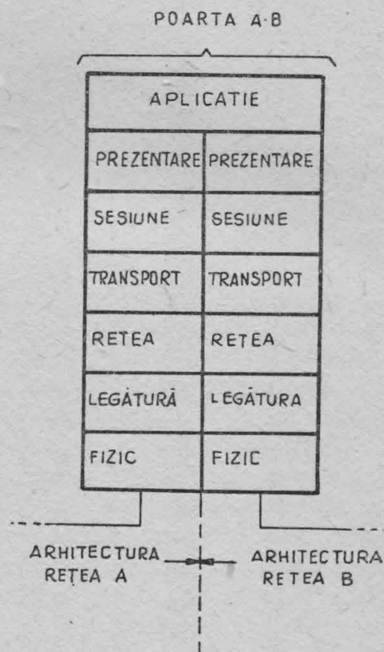


Fig. 9. Arhitectura de interconectare conform modelului OSI cu 7 niveluri.

Nu există nici un dubiu că sistemele informatice şi cele de conducere a proceselor tehnologice se vor dezvolta împreună şi că şi alte aplicaţii ale calculatoarelor (proiectarea asistată de calculator etc.) în mediul industrial se vor alătura acestora. Cerinţele pentru o arhitectură comună de reţea vor deveni stringente.

Actualmente se folosesc mai multe tipuri de reţele fizice pentru diferite aplicaţii (de exemplu, gestiunea economică, conducerea producţiei, supravegherea clădirilor, controlul accesului) care se suprapun geografic.

Figura 10 indică concepţia unui sistem universal de transport de mesaje care asigură transmiterea diferitelor tipuri de informaţii: similar sistemului telefonic, el poate deveni cu timpul o infrastructură standard într-o întreprindere. Toate subsistemele sînt conectate direct la această reţea care virtual nu va prezenta limitări de bandă de trecere. Astăzi

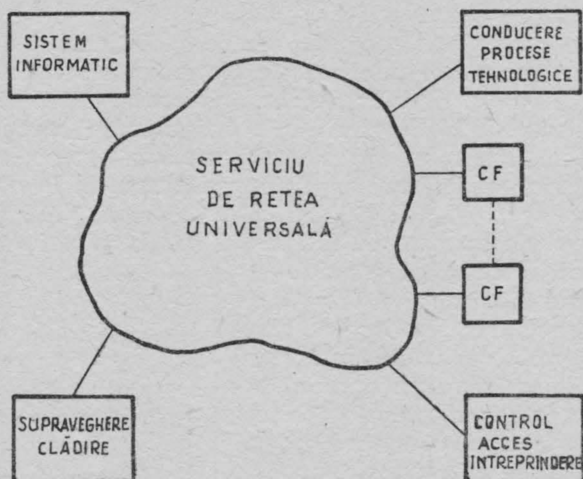


Fig. 10. Servicii multiple oferite de o reţea universală (CF — calculator Frontal).

sistemele de transmisie cu bandă largă (Cooper şi colab., 1983) pot constitui o soluţie, dar sistemele cu magistrală tip ETHERNET cu bandă sporită de transmisie (de exemplu, reţeaua SUPERNET cu o viteză de 100 Mbit/s sau reţele cu cablu cu fibre de sticlă şi viteze de transmisie de ordinul gigabiţilor/s) constituie de asemenea soluţii demne de luat în considerare.

Acest tip de reţea nestructurată pare a reprezenta o contradicţie faţă de structura ierarhizată logic a sistemelor de conducere a proceselor tehnologice şi a sistemelor informatice conform cu fig. 1. Dar căile fizice nu corespund obligatoriu cu cele logice: structura ierarhizată logic este explicată de incapacitatea noastră mentală de a stăpîni complexitatea fără un model de ierarhizare.

Deoarece toate prelucrările în timp real sînt deplasate spre calculatoarele frontale, cerinţele pentru sistemele informatice se aseamănă din ce în ce mai mult. În ambele domenii se vor utiliza aceleaşi tehnologii hardware-software, şi deci chiar marile întreprinderi nu-şi vor permite să suporte două organizaţii distincte responsabile pentru cele două categorii de sisteme de prelucrare automată a datelor şi probabil că unele din problemele interconectării celor două tipuri de sisteme evocate în prezentul material vor fi rezolvate printr-o redistribuire a responsabilităţilor şi a competenţelor.

Bibliografie

Biewald, J., ş.a. (1979). EPOS — A specification and design technique for computer controlled real-time automation systems. În *Proceedings of the 4th int. Conference on Software Engineering*, Munich 79. Springer, Berlin, pp. 245—250.

Bracker, W. E. (1983). Surveying the protocol conversion vendor's offerings. *Data Communications*, August 83.

Clocksin, W. F., şi C. S. Mellish (1981). *Programming in Prolog*. Springer, Berlin.

Cooper, E. B., ş.a. (1983). Design issues in Broadband local networks. *Data Communications*, Feb. 83, 109—122.

CSI Communications Solutions, Inc. (1983). *Access/SNA: Technical Overview*, 89—101.

DIN 66253 (1981) Teil 1. Informationsverarbeitung Programmiersprache PEARL *Basis PEARL:* (1982) Teil 2. Informationsverarbeitung Programmiersprache PEARL *Full PEARL*. Beuth, Berlin.

Färber, G. (1983). Konzept und Anwendung von Feldbussystemen mit verteilten Prozessperipherie-Modulen. În M. Syrbe und M. Thoma (Ed.), Interkama 83, *Fortschritte durch digitale Mess- und Automatisierungstechnik*. Springer, Berlin, pp. 495—510.

IBM (1983). Advanced control system ACS. *General information manual*.

IEEE (1981). 802 Local network standard. Draft B, Oct. 19.

Jenetzky, D., ş.a. (1983). Oberblick über das verteilte Automatisierungssystem Teleperm -M und Lösung Obertraögungsaufgaben mit dem Prozessbus CS275. În *KfK-PDV 224*, Kernforschungszentrum Karlsruhe, April 83. pp. 211—246.

Korn, G. A. (1977). Highspeed block-diagram languages for microprocessors and minicomputers in instrumentation, control, and simulation. În *Fachtagung Prozessrechner 1977*. Springer, Berlin, pp. 74—108.

MacWilliams, P. D., ş.a. (1984). Microcontroller serial bus yields distributed multispeed control. *Electronics*, Feb. 9, 125—130.

Mier, E. E. (1982). High-level protocols, standards, and the OSI reference model. *Data Communications*, July, 71—101.

Mostek (1982). SCU 20 Serial control unit, *Operation manual*.

NRC Network Research Corporation (1983). FUSION: *Users guide*.

Randesi, S. J. (1982). Interfacing minis and micros to IBM networks. *Mini-Micro-Systems*, March 82, 159—168.

Ritchie, D. M., ş.a. (1978). The UNIX-time-sharing System. *The Bell System Technical Journal*, 7/8. 78, 57, No. 6, Part 2, 1905—1930.

Walze, H. (1978). Bit serial communication in decentralized control systems — approaches for common solutions. *IFAC Conference* Helsinki.

Wiemann, B., Ries, W., Patz, M., Färber, G., şi Demmelmeier, F. (1981, 1982, 1983, 1984). rtp-Seminar Bussysteme. rtp *Regelungstechnische Praxis*, 23, 24, 25, 26.

XICOM (1983). The SNA Micro Node. *Product Information*.

P.4. SISTEM DE FABRICAȚIE FLEXIBIL DIN JAPONIA

Hiroyuki Yoshikawa

The University of Tokyo
Hongo 7-3-1, Bunkyo-ku, Tokyo 113, Japan

Articolul se ocupă cu dezvoltarea sistemelor flexibile de fabricație din Japonia. Progresul tehnologic în acest domeniu și proliferarea sistemelor au fost remarcabile în ultimii zece ani. În Japonia, în această perioadă, a fost derulat un Proiect național implicînd specialiști din institute de cercetare, universități și firme. Prima linie pilot apare acum ca rezultat al Proiectului; ea a influențat în mod evident tehnologia și concepția sistemelor flexibile de fabricație produse în acești zece ani. Conceptul de bază al acestui Proiect național este explicat în lucrare. El a rezultat dintr-o previziune asupra tendințelor de viitor ale dezvoltării economice, sociale și tehnologice. Este descrisă o concepție ideală potrivită cu previziunea, care a fost propusă la începerea Proiectului. Linia pilot care apare acum este fidelă Proiectului ideal în unele puncte dar nu și în altele. Ea constă din mai multe subsisteme, denumite complexe și fiecare din ele include — mai mult sau mai puțin — concepte de proiectare noi. Între ele, conceptul de structură complexă și metamorfism constituie trăsături caracteristice ale Proiectului. Comparația între linia pilot și Proiectul original reliefează unele puncte importante de rezolvat în viitor.

1. Introducere

În anul 1976, am avut șansa de a prezenta un articol (Yoshikawa, 1977) despre un Proiect japonez asupra metodologiei pentru fabricație fără oameni (Methodology for Unmanned Manufacturing = MUM) în care s-a efectuat o analiză a tendințelor automatizării în industria mecanică și s-au subliniat unele probleme de perspectivă. Concluzia articolului a fost că era extrem de important a se extrage o imagine ideală a viitoarelor sisteme de fabricație; altfel ar apărea situații de confuzii. În Proiectul MUM, existau multe discuții asupra imaginii viitoare a sistemelor, și s-au configurat multe concepte importante asupra fabricației, împreună cu unele proiecte conceptuale ale sistemelor.

În anul 1977 a fost implementat un Proiect Național, denumit Sistem de Fabricație Flexibil Complex cu Laser. Acest an, 1984, este anul final al Proiectului. O linie pilot apare acum în Japonia. Ea a adoptat multe concepte din Proiectul MUM. Proiectul MUM a demarat în 1972, deci au fost necesari 12 ani pentru a fi realizată o idee originală. Schimbările drastice în jurul Proiectelor pot fi subliniate din diferite puncte de vedere. În această perioadă, avansul și proliferarea sistemelor de fabricație flexibile a avut loc în diferite domenii ale industriei. Calculatoarele și alte tehnologii de bază, care sînt importante pentru ele, au fost îmbunătățite în mod remarcabil.

Conceptele de proiectare propuse în proiect, totuși, par încă juste și eficiente. De asemenea, FMS-urile comerciale dezvoltate într-o direcție bună de companiile japoneze au fost — fără dubiu — influențate de aceste concepte.

În prezenta comunicare, au fost extrase dintr-un scenariu concepțiile de proiectare și specificațiile tehnologice ale prototipului ce va fi descris ca și unele sisteme comerciale aflate practic în exploatare în industrie.

2. Scenariu pentru un sistem de fabricație viitor

Cum s-a menționat în ultimul capitol, au existat multe discuții în Proiectul MUM pentru configurarea unui sistem de fabricație viitor. În mod simplu, acele discuții sînt însumate în scenariul ce urmează, care este un mod de abordare metodic pentru caracteristicile ce derivă și pentru proiectul conceptual al unui sistem. Această abordare este dependentă de teoria proiectului general (Yoshikawa, 1977). Un sumar al rezultatului a fost prezentat în ultima conferință IFAC, ca un exemplu al abordării metodice a previziunii (Hatvany, 1982). Însă, pentru a explica filozofia proiectării, în legătură cu aspectele tehnologice ale sistemului prezentat, care va fi tratat în capitolele următoare, aci vor fi prezentate detaliile scenariului.

Ca o situație de fapt, au fost foarte dificil de schițat specificațiile pentru un sistem de fabricație care ar fi o tehnică de frontieră — încă în acest moment în ansamblul său .O previziune prudentă asupra condițiilor sistemului trebuie să fie făcută cu privire la factorii economic, social și tehnologic. Paragrafele următoare arată procesul obținerii schiței asupra specificațiilor.

2.1. Axiome

Să începem cu unele axiome. Axiome sînt, în mod uzual, propoziţiuni care sînt considerate corecte și care nu mai implică verificări. În cazul de față, factorii care afectează sistemul și nu sînt controlați sau modificați prin strategii sînt extrași ca axiome. Sistemul de producție trebuie să fie realizabil în mod economic și acceptat de oameni. De aceea, el este extrem de dependent de condițiile economice și sociale. Factorii extrași sînt următorii:

A_1. Criza de energie și de resurse naturale se va înrăutăți.

A_2. Concurența se va intensifica pe piețele lumii.

A_3. Numărul oamenilor extrem de instruiți și bine educați va crește.

A_4. Standardul de viață va continua că crească în mod rapid și va genera cerințe pentru o largă varietate de mărfuri.

Noi nu putem refuza să credem că totalul cantității de resurse naturale nu este infinit ci limitat, și — de asemenea — că se va ajunge la o criză serioasă dacă țările în curs de dezvoltare realizează un potențial de producție important cu consum de resurse și energie crescut în mod considerabil. De aceea, unele țări vor suferi — mai mult sau mai puțin — de pe urma crizei. În Japonia este destul de evident că se vor înrăutăți,

în mod serios condiţiile viitoare. Acest lucru nu poate fi schimbat prin vreo strategie.

Intensificarea concurenţei pe pieţele lumii nu poate fi evitată prin vreo strategie. Ne putem aştepta la o strategie prin care concurenţa va fi moderată pe piaţa internă, dar în lumea internaţională vor exista dificultăţi mai mari datorată incertitudinii în participare a ţărilor în curs de dezvoltare la piaţa mondială etc.

Există tendinţa generală ca nivelul mediu al educaţiei să crească în multe ţări. În mod special în Japonia compoziţia populaţiei se va modifica în mod rapid în maniera piramidei inversate. Acest fapt implică situaţia că ponderea muncitorilor neinstruiţi va descreşte, fapt intensificat treptat din cauza entuziasmului în creştere pentru educaţie. Am intrat într-o societate cu oameni „mai educaţi şi mai în vîrstă".

Standardul de viaţă a atins un nivel satisfăcător în ţările dezvoltate. În Japonia, în ultimii 20 de ani, standardul mediu a crescut iar prăpastia între bogaţi şi săraci a fost redusă în mod remarcabil, creîndu-se căi diferite de viaţă între oameni. Aceasta implică faptul că diferenţa individuală de viaţă va creşte şi vor fi generate cereri pentru o varietate largă de mărfuri.

2.2. Teoreme

Am încercat să obţinem unele propoziţiuni ce formulează condiţiile care afectează în mod direct sistemele de fabricaţie. Acest lucru va fi făcut pe o cale deductivă, astfel încît vom denumi aceste propoziţii *teoreme*.

1) Din A_1 *(criza de resurse şi energie)*, se estimează în mod direct că resursele şi energia sectoarelor consumatoare din industrie nu vor creşte foarte puternic în viitor. Japonia suferă de criză de teren, astfel încît în industriile cu suprafaţă mare creşterea productivităţii va fi binevenită. Poluarea va impune creşterea continuă a restricţiilor în viitor. De aceea, activitatea industrială trebuie să devină mai intensivă, mai concentrată şi mai eficientă în utilizarea resurselor. Aşa încît deducem următoarea teoremă:

T_1. *Importanţa relativă a industriei mecanice va creşte.*

Industriile siderurgice şi chimice, în prezent, de bază, îşi vor reduce rata lor de creştere, în comparaţie cu aceea a industriilor mecanice. Din motive similare cu cele de mai sus deducem o altă teoremă.

T_2. *Importanţa relativă a produselor cu prelucrări însemnate va spori.*

Acest fapt poate fi condiţia necesară pentru a realiza un profit mai ridicat cu consum de resurse mai mic.

2) A_2 *(Intensificarea concurenței)* va avea influență largă asupra caracteristicilor tehnice ale produsului, performanțelor de cost, fiabilității, mentenabilității etc. Există, deci, condiții necesare de supraviețuire pentru firme. Aceste condiții vor genera unele teoreme.

T_3. *Companiile vor face eforturi mult mai intensive pentru a dezvolta produse unicat.*

Aceasta va cauza în oricare companie, o creștere inevitabilă a investiției în cercetarea fundamentală, o proliferare a cercetătorilor și inginerilor de dezvoltare, reorganizarea structurii cercetării și dezvoltării etc. Costul produselor va fi unul din factorii cei mai importanți pentru supraviețuire, de aceea obținem teoremele T_4, T_5, T_6, de asemenea strîns legate cu problemele de resurse (A_1).

T_4. *Îmbunătățirea productivității muncii va deveni cu mult mai importantă.*

T_5. *Durabilitatea produselor trebuie să crească.*

T_6. *Va exista o reducere drastică a perioadelor de întîietate.*

Dacă noi comparăm productivitatea muncii din industria mecanică cu aceea din industriile siderurgică și chimică, prima trebuie să fie mult mai scăzută decît ultima, din cauza automatizării mai scăzute în prima. Totuși, importanța relativă a industriei mecanice este în creștere, de aceea va fi esențială necesitatea creșterii productivității sale. Acest lucru va implica îmbunătățirea substanțială în tehnicile industriilor mecanice.

3) A_3 (creșterea muncitorilor bine educați și mai în vîrstă) va implica, de asemenea, schimbări tehnologice. Serios vorbind, este un fapt idealizat ca cererea și oferta de forță de muncă să coincidă cantitativ și calitativ. Cu alte cuvinte, industria este solicitată să ofere locuri de muncă cum doresc oamenii. În concordanță cu A_3, populația muncitoare are tendința pentru mai educat și mai în vîrstă, deci putem deduce următoarea teoremă:

T_7. *Industria va fi obligată să ofere locuri de muncă mai intelectuale, atît ca număr cît și ca nivel.*

„Meritul" vîrstei este — în general — creșterea priceperii, care nu se va deteriora atît de mult cu vîrsta, cum este cazul capacității fizice; în același timp, activitățile intelectuale vor fi solicitate de oameni bine educați. Se obține o altă teoremă, în același mod ca mai sus.

T_8. *Vor exista cerințe sporite pentru condiții de lucru mai confortabile.*

Această teoremă sugerează că vor fi bine primite locurile de muncă cum sînt cele de *tip birou* — funcționăresc, mai curînd decît cele de *tip fabrică*, locuri eliberate de regimul de lucru în tactul mașinii și în care oamenii au posibilitatea de a-și îmbunătăți calificarea.

4) Din A_4 *(Standard de viață)* vor fi deduse două teoreme.

T_9. *Veniturile sporite creează cerințe sporite pentru servicii.*

Cînd veniturile populației cresc, există tendința de a se scurta timpul de lucru pentru a se obține mai mult timp de odihnă; dar, în special în Japonia, acest lucru nu se va petrece. Din contră, omul va încerca să cîștige mai mult, pentru cumpărarea mai multor servicii. Serviciile vor fi

ușor disponibile pentru a îmbunătăți calitatea vieții. Va rezulta că valoarea totală a muncii umane nu va descrește, chiar după ce este realizată o productivitate a muncii însemnată în industria mecanică. Dacă privim la calitatea produselor realizate în această industrie, pentru a deduce ultima teoremă, constatăm următoarele:

T_{10}. *Vor spori cerințele pentru varietate mai largă de produse, așa încît diversificarea produselor va deveni intensivă.*

2.3. Leme

Pasul următor în prezenta procedură este de a stabili lemele: condițiile care specifică în mod direct sistemele de producție viitoare. Cu alte cuvinte, acestea alcătuiesc un set de condiții și caracteristici tehnice funcționale pentru sistemele de fabricație viitoare ce urmează a fi construite în Proiect. În mod direct, din A_1, care vorbește despre criza de energie și resurse, obținem:

L_1. Un sistem de producție viitor trebuie să fie un sistem cu economie de energie și resurse, care implică un sistem de fabricație minimal pentru producțiile date. Apoi, din T_2 (produse cu grad de prelucrare ridicat) și T_5 (durabilitatea produselor) se deduce următoarea lemă:

L_2. Sistemul trebuie să fie capabil de a îndeplini sarcini sigure și deosebit de complicate. Din T_3 (produse unicate), T_7 (activități intelectuale) și T_8 (condiții de lucru confortabile) este obținut:

L_3. Sistemul trebuie să fie capabil a utiliza priceperea oamenilor cu calificare înaltă, pe o cale satisfăcătoare și să fie capabil de a comunica cu ei la un nivel adecvat. Din T_4 (productivitatea muncii și T_6 (reducerea perioadei de întîietate) se obține:

L_4. Sistemul, ca un tot, trebuie să aibă o productivitate a muncii înaltă, ca răspuns rapid la cerințele producției. Din T_8 (condiții de lucru confortabil) este dedusă următoarea lemă:

L_5. Sistemul trebuie să satisfacă condiția de separare strictă a muncitorilor și mașinilor în spațiu și în timp. Din T_9 (servicii) și T_{10} (varietate de produse), este dedusă ultima lemă:

L_6. Sistemul trebuie să fie deosebit de flexibil.

3. Proiectarea conceptuală a viitorului sistem de fabricație

Prin considerarea lemelor obținute în ultimul capitol, vom schița un proiect conceptual al viitorului sistem de fabricație. Lemele constituie caracteristici tehnice funcționale pentru acest sistem. În general vorbind, nu există metodă deterministă pentru a dezvolta un proiect dacă sînt date caracteristicile tehnice funcționale. Teoria proiectului general (Yoshikawa, 1981) arată că elaborarea caracteristicilor tehnice poate să ne ajute în a rezolva această problemă. În următoarele aliniate, fiecare lemă va fi discutată în detaliu și vor fi ilustrate următoarele realizări de funcțiuni particulare.

3.1. Structura complexă pentru economia de resurse şi energie (L_1)

Dacă cercetăm această lemă, constatăm că sistemul trebuie să conţină numai părţi care sînt în mod direct necesare procesului de producţie. Totodată, spaţiul ocupat trebuie să fie minim.

Analiza actualului sistem de producţie ne face să luăm notă despre faptul că se menţine un volum important de transport al materialelor şi pieselor. Acest transport nu contribuie la producţie, în sensul strict al activităţii de producţie.

Atunci, cum putem elimina acest transport? Ca una din metode pentru a-l realiza, o idee a fost propusă în Proiectul MUM, respectiv o Structură Complexă, aşa cum se arată în Fig. 1. În liniile automate de producţie obişnuite există puncte de prelucrare individuale ca: strunjire, frezare, control şi asamblare. Corespunzător, ele necesită dispozitive particulare de încărcare şi descărcare, care leagă punctele de lucru şi echipamentele de transfer. În conceptul structurii complexe, toate operaţiile sînt integrate într-un singur spaţiu,aşa cum s-a ilustrat în Fig. 1, şi astfel toate punctele de lucru sînt instalate astfel încît să se interfaţeze în mod corespunzător. Orice piesă odată cuplată într-o mandrină este reţinută cuplată cît mai mult posibil.

Dacă este necesar, o piesă poate fi transferată dintr-o mandrină în alta, direct, fără utilizarea vreunui echipament de transfer. Ca rezultat, putem elimina utilajele de transfer şi de încărcare-descărcare care nu contribuie la fabricaţie. Această structură are şi un merit, remarcabil, în reducerea suprafeţei de producţie necesare. Pe lîngă aceasta, poate cel mai important lucru este că se elimină timpul de transfer, reducînd timpul total de producţie.

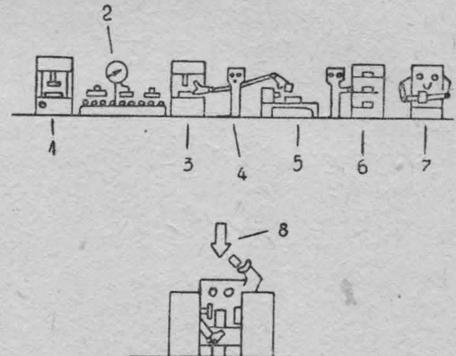


Fig. 1. *Concept al Structurii Complexe:* 1) uzinare; 2) control calitate; 3) uzinare; 4) transfer şi încărcare; 5) uzinare; 6) depozit; 7) asamblare; 8) integrare.

3.2. Integrarea diferitelor operaţii pentru solicitări complicate (L_2)

La sistemele viitoare, sînt necesare solicitări complicate, pentru a fabrica produse cu valoare ridicată. În acest scop, am conchis că sistemul trebuie să fie capabil de a realiza operaţii complexe cum ar fi: sudare, tratament termic, tratamente de suprafeţe şi asamblarea pieselor într-un produs complex; vor fi necesare operaţii de control şi verificare, atît pentru părţile componente cît şi pentru produsul finit. Această cerinţă este contradictorie cu structura complexă, fiindcă spaţiul singular din această

structură va crește, în mod inevitabil, în proporție cu creșterea numărului de operații specifice. Dar acest aspect va putea fi rezolvat în viitor, prin soluții tehnologice noi.

3.3. Comunicația avansată om-mașină pentru condițiile de mediu de tip „intelectual" (L_3)

Această cerință este strîns legată cu sistemele de programe de tip CAD/CAM (Computer Aided Design/Computer Aided Manufacturing = Proiectare Asistată de Calculator/Fabricație Asistată de Calculator). În viitor, sistemele CAD/CAM vor deveni foarte „inteligente", prin utilizarea tehnicilor inteligenței artificiale și de inginerie a cunoștințelor (Tomiyama, 1984). Aceasta determină ca oamenii să poată fi angajați în preocupări intelectuale foarte ridicate.

Informația, ca rezultat al procesului de gîndire umană, trebuie să fie introdusă — în mod eficient — într-un sistem. Ea poate fi abstractă și complicată. Ar putea fi utilizate, pentru acest scop, limbaje de nivel înalt sau, posibil, limbajul natural. De exemplu, pentru sisteme de producție ca prezentul Proiect, va fi necesară o muncă de proiectare, de valoare considerabilă, din cauza flexibilității sale. Majoritatea componentelor Proiectului nu va implica, totuși, originalitatea ridicată a proiectanților; în schimb, modificările parțiale și schimbările dimensionale pot constitui majoritatea activităților de proiectare. În cazul unui sistem CAD inteligent, proiectanții nu sînt solicitați să introducă fiecare schimbare dimensională în sistem. Ei pot să introducă numai modificările principale, sistemul modificînd în mod automat alte părți necesare ale Proiectului, stabilind soluția mai eficientă (Kitajima, 1984).

În asemenea sisteme, proiectanții sînt solicitați să introducă mai curînd informații funcționale decît date geometrice. Introducerea datelor pe cale vocală, printr-un limbaj de nivel înalt, similar cu limbajul natural, este, de exemplu, de dorit.

3.4. Sistem bine controlat și sigur pentru productivitate înaltă (L_4)

Pentru a spori productivitatea muncii, extinderea automatizării este mijlocul cel mai eficace; pe această cale se reduce, în mod direct, numărul de muncitori și, deci, se obține o productivitate a muncii ridicată. Automatizarea este, totuși, dependentă de alți factori, în special de activitățile pregătitoare, sau de suport, cum este întreținerea și producția de software. Ambele influențează puternic gradul de utilizare a întregului sistem. Productivitatea producției de software poate fi îmbunătățită, de exemplu, prin folosirea unui limbaj de nivel înalt.

Dar, în mod fundamental, este mai bine să se reducă volumul de informații. Primul deziderat va fi realizat prin promovarea CAD/CAM, iar un exemplu a fost arătat în ultima secțiune, adică un CAD inteligent. Pe lîngă aceste sisteme CAD inteligente sînt așteptate sisteme noi, care asigură generarea automată a planurilor de producție, cu o funcție de

recunoaştere a caracterelor şi o bază de date care să permită planificarea automată a producţiei în funcţie de starea de funcţionare a fabricii. Aceste probleme vor fi rezolvate prin îmbunătăţirea tehnicilor de elaborare a software-ului necesar.

Din contră, ultimul deziderat, reducerea însăşi a informaţiei necesare, este funcţie de proprietăţile hardware ale sistemului de fabricaţie. De exemplu, să considerăm un tip de sistem de fabricaţie obişnuit pentru preocupările unui atelier care are în el un număr considerabil de maşini-unelte; dificultatea menţinerii unui coeficient ridicat de utilizare a acestor maşini-unelte creşte proporţional cu numărul lor. Astfel, dificultăţile privind prelucrarea informaţiei cresc foarte mult. Metoda „primul venit-primul servit" (FIFO) poate elimina această dificultate, dar, în mod normal, se pierde din productivitate, din cauza pierderii de timp între operaţii. Printr-o proiectare specială a maşinii-unelte este posibil să se realizeze un compromis pentru rezolvarea acestei contradicţii. Proiectul care va fi explicat în secţiunea următoare (3.6) poate fi un răspuns la această problemă.

Întreţinerea sistemului este o problemă mult mai serioasă. Timpii de cădere datorită defecţiunilor sistemului vor înrăutăţi în mod direct productivitatea. Pentru îmbunătăţire, fiabilitatea mai ridicată trebuie implementată prin Proiectul sistemului, iar întreţinerea mai bună trebuie realizată prin îmbunătăţirea tehnicilor de diagnoză şi reparare.

Diagnoza automată şi repararea vor fi tehnici puternice pentru îmbunătăţirea fiabilităţii totale a sistemului. Dezvoltarea unor asemenea tehnici nu a fost inclusă în Proiect, dar o seamă de alte proiecte au fost implementate în Japonia referitor la automatizarea diagnozei şi reparării, urmînd a fi aplicate nu numai în industria mecanică ci şi în alte domenii. Interesul deosebit al acestor proiecte este de a dezvolta roboţi „inteligenţi", care pot realiza activităţi de diagnoză şi reparare a utilajelor mecanice, a instalaţiilor chimice, a centralelor nucleare, a structurilor subacvatice şi a altor sisteme automate. Cel dintîi proiect a fost iniţiat de noi în departamentul utilajelor de precizie, Universitatea din Tokyo (Yoshikawa, 1980). Proiectul a demarat în 1979 şi a investigat trăsăturile caracteristice ale lucrărilor de diagnosticare şi reparare în diferite domenii, lucrări care în mod normal sînt executate manual.

Au fost stabilite unele cerinţe funcţionale pentru un asemenea robot, denumit robot de întreţinere:

1) Afinitate bună a manipulatorului faţă de obiectele din întreţinere care au o structură regulată, ca de exemplu un drum în trepte.

2) Dexteritate ridicată a manipulatorului care penetrează şi operează la punctul de întreţinere.

3) „Inteligenţă" avansată în a înţelege stadiul obiectivelor de întreţinere şi pentru a genera planul de întreţinere în concordanţă cu acest stadiu.

Recent, alte cîteva proiecte au fost implementate. Unul din ele, denumit JUPITER, este cel mai mare şi a fost suportat de către MITI.

3.5. Fabrica fără oameni, pentru separarea muncitorilor de mașini în spațiu și timp (L_5)

Conceptul separării muncitorilor și utilajelor în timp și spațiu a fost propus inițial de către Williamson în cadrul prezentării asupra Systemului 24 din anul 1967 (Williamson, 1967). Conceptul a devenit mai important acum, deoarece din ce în ce mai puțini muncitori doresc să acționeze utilajele direct în fabrici. Soluția clară pentru această cerință este de a construi o fabrică fără oameni, dotată cu o cameră de comandă cu oameni, cameră care este curată și confortabilă pentru operatorii care lucrează aici; acești operatori pot fi angajați în planificarea și programarea procesului de fabricație al fabricii fără oameni.

3.6. Metamorfism pentru flexibilitate mai ridicată (L_6)

Flexibilitatea este principala caracteristică a sistemelor viitoare, respectiv este principala motivație pentru realizarea prezentului Proiect. Să reamintim istoria dezvoltării în automatizarea flexibilă. Mașinile-unelte convenționale care erau acționate manual dispuneau de o flexibilitate considerabilă. Automatizarea a sacrificat această flexibilitate; strungurile automate și liniile de transfer fiind exemple tipice. Introducerea comenzilor numerice a recîștigat, cu succes, flexibilitatea mașinilor-unelte, fără pierderea nivelului înalt de automatizare. Combinația calculatoarelor și a acestor mașini-unelte a îmbunătățit simțitor flexibilitatea, pentru a o atinge pe aceea a mașinilor-unelte convenționale manuale.

În mod fundamental, aceste mașini-unelte comandate numeric sînt dotate cu flexibilitate, prin flexibilitatea software-ului. Dar există o limită a flexibilității, datorită structurii mașinii: un strung nu poate opera bine pentru fabricarea pieselor de mașini prismatice, deoarece are o structură care nu este bună pentru generarea suprafețelor plane. Pentru a face față acestei limite, trebuie să echipăm tipuri diferite de mașini-unelte, ca strunguri, mașini de frezat, mașini de găurit etc. Pentru o cerință particulară a fabricației este stabilită, în mod uzual, combinația optimă a acestor mașini-unelte. Cu alte cuvinte, în cazul cînd cerințele producției se modifică optimalitatea va fi pierdută. În mod normal, sculele și cuțitele pot fi schimbate pentru fabricația de componente de diferite forme, dar flexibilitatea poate fi crescută, în continuare, numai prin modificarea modulelor mașinii-unelte însăși. În acest caz, nivelul flexibilității poate fi categorisit pe baza tipurilor de module ce urmează a fi schimbate, respectiv:

1) Scule și cuțite.

2) Suport sculă/cuțit.

3) Mișcare mecanisme de comandă.

4) Acționări.

5) Ghiduri de mișcare.

6) Paturi (batiuri).

Cele mai obişnuite maşini-unelte în prezent sînt cele de la nivelele 1) sau 2). Un exemplu arătat în fig. 2, poate fi categorisit în nivelul 3). În caz că unităţile schimbabile din figură conţin acţionări specifice, atunci se ajunge la nivelul 4). Un exemplu al celui mai înalt nivel de flexibilitate este cel prezentat în fig. 3: se schimbă structura patului şi fiecare unitate este standardizată şi corespunde unei clase de maşini; unele configuraţii pot produce un strung, altele pot produce o maşină de frezat, deci obţinem flexibilitatea cea mai ridicată.

Metoda de a îmbunătăţi flexibilitatea funcţională prin modificarea structurii este denumită *metamorfism*. Există multe probleme tehnice de rezolvat pentru a desăvîrşi o asemenea metodă, dar acest lucru este foarte eficace pentru îmbunătăţirea flexibilităţii funcţionale a sistemelor de fabricaţie.

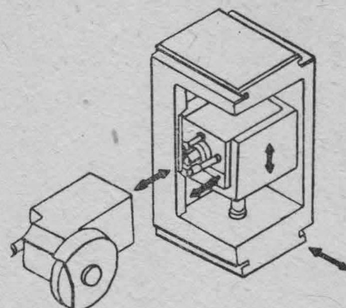


Fig. 2. *Sistem metamorfic* (a).

4. Linia pilot a proiectului

Cum s-a menţionat în primul capitol, o linie pilot a fost deschisă publicului în iulie 1984 în Laboratorul de Inginerie Mecanică — Tsukuba. În acest capitol se va face o scurtă prezentare a acestei linii.

4.1. Cerinţe funcţionale

Obiectivul construirii liniei pilot este de a confirma valabilitatea şi fezabilitatea unui asemenea sistem propus în Proiect, pentru obţinerea diferitelor date prin experimentări de produse din fabricaţia curentă. Produsele fabricate pe această linie vor fi de trei feluri: reductoare cu două trepte, reductoare cu trei trepte şi unităţi pentru axul principal al maşinii-unelte.

Funcţia liniei pilot este categorisită în şase subfuncţiuni astfel:

1) prelucrarea materialelor brute;
2) uzinare;
3) asamblare;
4) prelucrare prin laser;
5) inspecţie;
6) planificarea proiectării şi producţiei.

Aceste funcţiuni au fost obţinute în linia de la Tsukuba. Iniţial prelucrarea materialului brut s-a realizat în mod separat, într-un laborator diferit, din cauza gabaritului limitat al liniei. Produsele liniei pilot de

prelucrare a materialului brut vor fi livrate către linia de la Tsukuba, care include funcţiile 2)—6). Funcţia liniei pilot acoperă o gamă largă de activităţi de fabricaţie; ea depăşeşte conceptul normal al unui sistem flexibil de fabricaţie (FMS) şi de aceea, denumim prezentul sistem un Sistem Complex de Fabricaţie Flexibilă (FMSC — Flexible Manufacturing System Complex).

Cerinţa pentru eficienţă a prezentului sistem a fost stabilită în mod simplu, astfel: timpul de fabricaţie a oricărui produs trebuie să fie sub 50% din timpul de fabricaţie actual. Aspectul economic nu a constituit un interes major la începutul Proiectului, fiindcă a fost, în mod practic, imposibil de a estima costul unui asemenea sistem ce ar apărea pe piaţă după zece ani. Tehnicile şi tehnologiile de realizare vor fi mult dezvoltate, costul materialelor s-ar schimba etc. De aceea am considerat numai timpul de producţie, pentru a evalua sistemul viitor la începutul Proiectului.

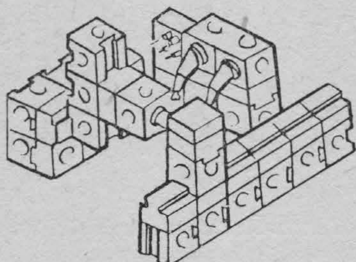


Fig. 3. *Sistem metamorfic* (b).

4.2. Sistemul FMSC total

Sistemul FMSC total este reprezentat în fig. 4 (Ozaki, 1983). Blocul din josul figurii ilustrează sistemul hardware al FMSC, care constă din cinci subsisteme: sistem de prelucrare a materialului brut, complex de prelucrare, complex de asamblare, complex de inspecţie, complex de prelucrare prin laser.

Partea superioară a fig. 4 arată schema bloc a prelucrării informaţiei pentru generarea instrucţiunilor de acţionare a sistemului hardware: în partea stîngă se găseşte prelucrarea informaţiilor pentru generarea datelor necesare produselor care se prelucrează, iar în partea dreaptă este arătată prelucrarea informaţiilor necesare controlului fabricaţiei, incluzînd decizia de planificare a celorlalte faze metamorfice etc. Ambele fluxuri ale informaţiei vin în blocul de control al procesului, care asigură — în mod direct — instrucţiunile operaţionale la fiecare unitate de acţionare a sistemelor hardware.

Planul subsistemelor, 2)—6) din secţiunea 4.1, este arătat în fig. 5. Conceptul de „complex" este realizat numai în interiorul fiecărui subsistem. Cu alte cuvinte, există un sistem de vehiculare automată convenţională care transportă materiale, scule şi fabricate între subsisteme. Iniţial s-a discutat, de exemplu, ca funcţiile de uzinare şi asamblare să fie integrate într-un spaţiu comun, pentru a realiza un complex de prelucrare-asamblare cu eficienţă mai ridicată. S-a constatat că un asemenea

nivel ridicat de complexitate nu este fezabil, din cauza necesităţii unui spaţiu de lucru prea mare, care să asigure ambele operaţii. Schema din fig. 5 are, însă, cîteva avantaje importante: eliminarea transferului între operaţiile dintr-un subsistem, pentru diminuarea timpului de fabricaţie; faptul că necesitatea transferului este limitată la intersubsisteme înseamnă că fluxuri de materiale, piese şi fabricate au loc numai de-a lungul direcţiei de înaintare.

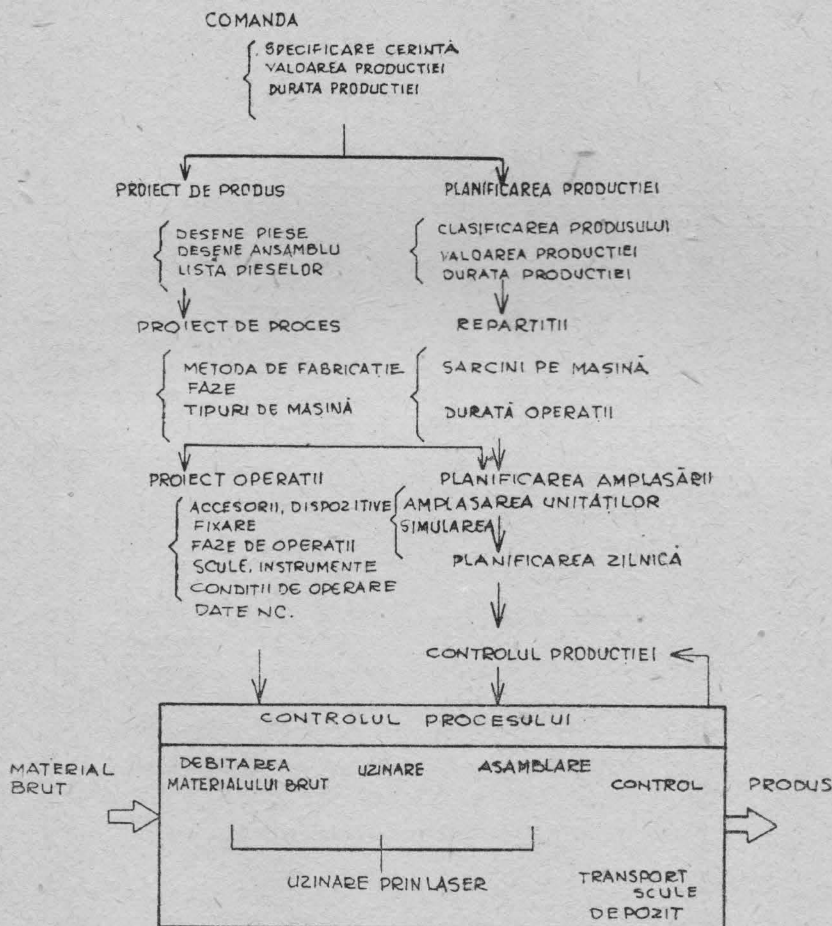


Fig. 4. *Schema bloc a sistemului general FMSC.*

Toate acestea fac ca structura întregului sistem să fie foarte simplă şi să uşureze sarcina sistemului de programe.

4.3. Complexul de prelucrare (uzinare)

Complexul de prelucrare este ilustrat în fig. 6 (Karita, 1983) și constă din trei celule de prelucrare, care au funcțiile următoare:

Celula nr. 1: 1. Mișcare laterală a coloanei (axa X).
2. Rotația coloanei (axa B).

Celula nr. 2: 1. Mișcarea laterală a coloanei (axa X).
2. Mișcarea înainte și înapoi a coloanei (axa Z).
3. Mișcarea verticală a traversei (axa Y).

Celula nr. 3: 1. Mișcarea înainte și înapoi a coloanei (axa Z).
2. Mișcare verticală a traversei (axa Y).

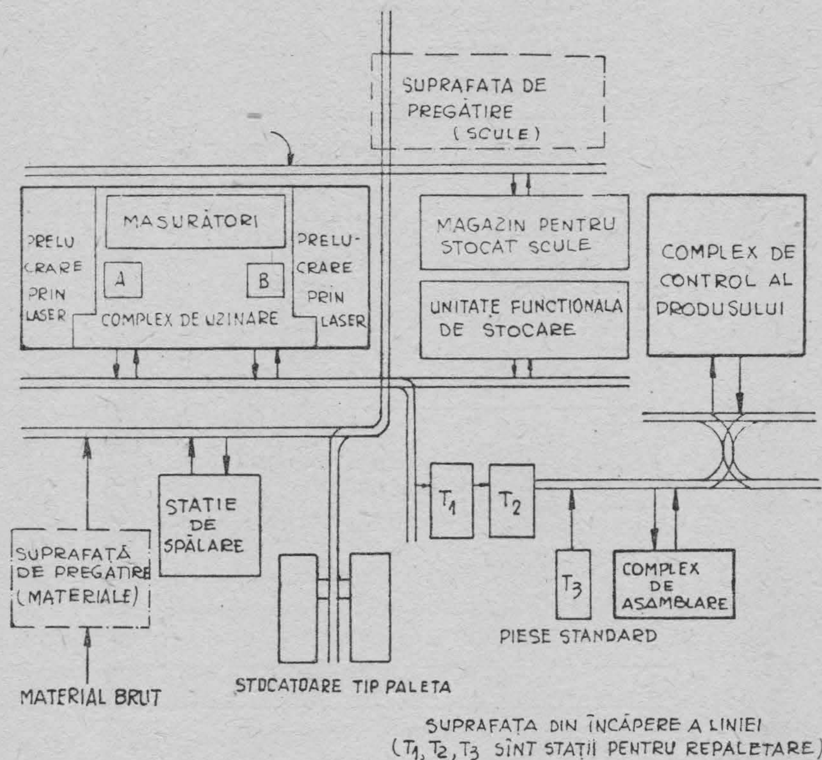


Fig. 5. *Amplasarea liniei pilot.*

Combinarea acestor mișcări va face față operațiilor de prelucrare cerute pentru piesele date, care sînt mai mici de 300 mm în oricare direcție. Există unități de prelucrare standardizate, care pot fi instalate pe orice coloană (metamorfism). Însă, ele au funcțiuni diferite, precum urmează:

1) Unitate principală de rotație, care susține și rotește piese de prelucrat cilindrice și cuțite rotative.

2) Unitate de indexare, care susţine scule staţionare şi piese de prelucrat prismatice pe care le indexează.

3) Unitate care strînge şi desface şuruburi, respectiv înserează şi extrage bolţuri.

4) Unitate de rotaţie cu viteză ridicată, care asigură operaţii de mare viteză, cum sînt rectificarea (de la 20 rot/min la 20 000 rot/min).

5) Unitate de ajustare a cuţitului, care ajustează poziţia cuţitului de găurire în mod automat.

Configuraţiile posibile ale complexelor sînt arătate în fig. 7. Se vede uşor, din această figură, că flexibilitatea este foarte ridicată, iar productivitatea este — de asemenea — îmbunătăţită, prin execuţia concurentă a mai multor operaţii.

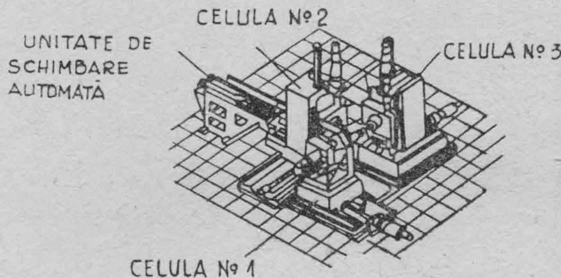


Fig. 6. *Complex de uzinare.*

	CONFIGURAŢIA CELULEI	FUNCŢIA
1		Uzinare pe două feţe Plăci şi discuri
2		Uzinare pe patru feţe Piese cilindrice şi prismatice
3		Uzinare pe şase feţe Uzinare simultană pe două feţe Piese cilindrice şi prismatice
4		Uzinare pe şase feţe Uzinare simultană pe trei feţe
5		Uzinare simultană a două piese

Fig. 7. *Configuraţiile complexului.*

Trebuie subliniat faptul că sistemul cu această concepţie a fost realizat pe baza unor cercetări asupra tehnicilor de bază, dintre care unele sînt listate în continuare:

1. Tehnici de prelucrare complexă.
2. Tehnici de prelucrare mecanică.

3. Analize termice pentru structuri modulare.
4. Mecanisme din structuri modulare.
5. Tehnici de frînare.
6. Suporturi de piese de prelucrat pentru uz general.
7. Diagnoza defectului.
8. Tehnici de asigurare a preciziei.

Linia pilot care a rezultat este fructul acestor cercetări de bază.

4.4. Complex de asamblare

Asamblarea automată este o tehnică tînără între tehnicile de fabricație. În domeniul producției de masă, ca de exemplu în industria producătoare de ceasuri și în industria automobilelor, s-a realizat un oare-

Fig. 8. *Un exemplu de amplasament al sistemului complex de asamblare:* 1) un exemplu de amplasament al sistemului complex de asamblare; 2) casetă de piesă; 3) unitate de apucare; 4) unitate de inserție și interferență; 5) unitate de strîngere pentru șuruburile conducătoare; 6) scală de asamblare; 7) stocator pentru scule de asamblare; 8) schimbător automat de scule; 9) a doua stație de asamblare pentru șuruburi conducătoare; 10) prima stație de asamblare pentru accesorii; 11) unitate de transfer piesă; 12) piesă de asamblat; 13) sistem complex de uzinare.

care grad de automatizare al lucrărilor de asamblare, înainte de recenta epocă a roboților. S-au realizat mașini pentru scopuri speciale, în care fiecare realizează o operație simplă, particulară, cum ar fi introducerea de pini în găuri de dimensiuni fixe; fiecare operație din munca de asamblare a necesitat mașini-unelte de asamblare automată, proiectate în mod particular: strung, mașină de frezat etc.

Îmbunătățirea rapidă recentă a automatizării asamblării a fost susținută prin progresul remarcabil al roboților industriali. Ei au devenit, în mod gradat, o tehnică curentă pentru asamblare și s-au dezvoltat multe tipuri de roboți de asamblare în diferite domenii ale industriei. Pare însă prea devreme pentru a fixa categoriile lor; posibilitatea progresului viitor al roboților este foarte ridicată și este de așteptat ca un tip complet nou de robot de asamblare să apară în mod frecvent.

Aceste fapte conduc la concluzia că ideia metamorfismului, care poate fi definit ca tranziția ușoară de la o categorie la alta, poate avea o valoare mai mică decît complexul de prelucrare, deoarece categoria însăși nu poate fi în mod clar stabilită în cazul asamblării. Deci, în acest Proiect, nivelul metamorfismului a fost limitat doar la nivelul sculei, menționat în secțiunea 3.6.

Structura complexului de asamblare este reprezentată în Fig. 8 (Hitano, 1983). El are un pat cu un cadru rigid, pe care este montat un suport de scule mobil. Sînt pregătite tipuri diferite de scule. Aceste scule sînt proiectate, în particular, pentru operații de asamblare. Multe din ele sînt asemănătoare „mîinilor de roboți" atașate pe manipulatoare. Din nefericire, a fost imposibil să se dezvolte o „mînă atotputernică", care să îndeplinească toate operațiunile cerute în operațiile de asamblare. Unele exemple de „mînă" utilizată pentru asamblarea unei unități de rotire sînt arătate mai jos:

1) Introducerea manșoanelor;
2) Acționarea șuruburilor;
3) Alimentarea bilelor;
4) Fixarea protecțiilor (capacelor);
5) Manipularea axelor.

Fiecare din aceste „mîini" are o formă specială și un mecanism adecvat pentru a realiza operația particulară necesară.

4.5. Complex de control al produsului

Controlul produsului realizat îmbracă o mare varietate de soluții depinzînd — în primul rînd — de tipul funcțiunilor produselor. Construcția unui sistem flexibil de control este o problemă controversată și dificilă; controlul fiecărui produs este mult mai important în automatizarea flexibilă decît în producția de masă din cauza următoarelor probleme:

1) Adesea este necesar a atașa produsului toate datele controlului.
2) Nu este disponibilă o statistică asupra relației între precizia piesei și funcțiunile produsului.
3) Erorile de fabricație neașteptate și nedetectate pot avea loc relativ mai frecvent decît în producția de masă, din cauză că o piesă

a sistemului de fabricație poate acționa, mai puțin frecvent, la un timp oarecare și după o lungă pauză.

4) Din cauza structurii complicate a sistemului de fabricație, nu este economic să se instaleze un număr mare de instrumente de diagnoză în sistem. În schimb, verificarea funcționării produsului final, care este reflectarea sistemului, poate, uneori, da informații utile asupra defecțiunilor sistemului.

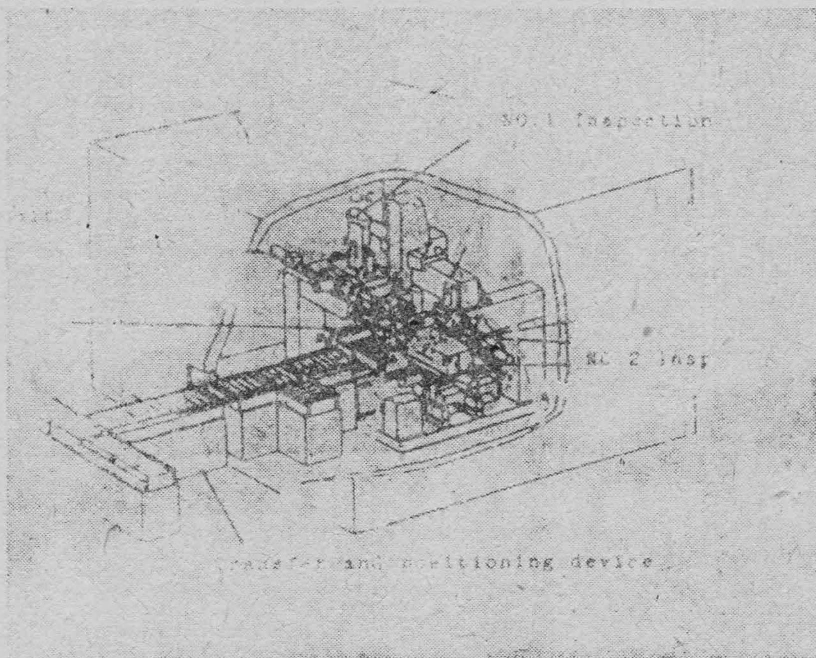


Fig. 9. *Vedere generală a complexului:*
1) structura instalației de test; 2) unitatea de control nr. 3; 3) produsul testat; 4) dispozitiv de poziționare și transfer; 5) unitate de control nr. 2; 6) unitate de control nr. 1.

Deci, în prezentul Proiect a fost dezvoltat un complex de control automat a produsului care se prezintă în Fig. 9 (Higashimoto, 1983). Sistemul este echipat cu palete standard pe care se poate fixa orice tip de produs. Punctele de control asupra produsului sînt stabilite anterior în locul în care accesul traductoarelor este asigurat în mod automat. Măsurătorile realizate, inclusiv precizia acestora, sînt listate mai jos:

1) Lungime (± 3 µm/300 mm);
2) Formă (± 5 µm/300 mm);
3) Eroarea statică ($\pm 1{,}5$ µm);
4) Eroarea dinamică ($\pm 1{,}0$ µm);
5) Vibrația ($\pm 0{,}5$ µm sau $\pm 0{,}01$ g/10+5 000 Hz);
6) Sgomot ($\pm 1{,}5$ dB/30—500 Hz);

7) Temperatura ($\pm 1°C/30°C$);
8) Momentul ($\pm 0{,}2$ kg·cm);
9) Mișcare în gol a reductorului ($\pm 10''$).

Aceștia sînt parametrii de bază măsurați prin traductoare specializate. Rezultatele acestor măsurători sînt transmise la un calculator, al cărui sistem de diagnosticare stabilește gradul de acceptabilitate a produsului respectiv; în cazul în care produsul nu este acceptat, sistemul de diagnosticare deduce punctul de eroare din sistemul de fabricație, pe baza datelor măsurate anterior.

4.6. Laser

Laserul constituie baza unei metode puternice pentru prelucrare mecanică, care are următoarele avantaje față de alte metode:

1) Laserul este utilizat pentru a realiza practic orice formă (sculă universală).
2) Diferitele tipuri de proces, ca rectificarea, frezarea, sudura, tratarea suprafeței etc. sînt conduse prin același echipament cu fascicul laser.
3) Laserul este aplicabil pentru procese de prelucrare din materiale diverse; sînt acceptate metalele, ceramicile, plasticele, lemnul.
4) Transmisia fasciculului de laser este ușoară.
5) Este posibilă utilizarea simultană a fasciculului laser atît pentru prelucrare cît și pentru măsurări.
6) Nici o forță de reacție nu intervine la fixarea foarte ușoară a materialelor.

În Proiect au fost realizate unele tipuri de laser cu CO, cu putere maximă 20 kW. De asemenea, laserul Ar de 200 W și laserul YAG de 300 W au fost dezvoltate pentru prelucrarea materialelor cu precizie ridicată.

5. Probleme de viitor

5.1. Evaluarea FMSC

Evaluarea FMSC-ului este o problemă de viitor, însă un aspect general va fi prezentat în continuare. Așa cum s-a menționat anterior, o evaluarea economică la acest stadiu este dificilă, din cauza diferiților factori care afectează realizarea practică a liniei. Prin urmare, evaluarea flexibilității și a timpului de fabricare poate fi posibilă numai dintr-un punct de vedere tehnologic. Investigarea timpului de fabricare a început acum, prin utilizarea FMSC-ului aflat în funcțiune. Va fi necesar un timp mai mare pentru a obține o înțelegere deplină a rezultatelor. Vom discuta în continuare despre flexibilitate. Ce este flexibilitatea?

Nu este ușor să se dea o definiție a sistemului flexibil de fabricație. Pe scurt, putem spune că un FMS nu este un sistem de producție de masă; sistemul de producție de masă este definit ca fabricația numai a unui singur tip de produs iar sistemul FMS fabrică mai mult decît un

produs: de la două la un număr nelimitat. Trebuie să recunoaştem că există o varietate largă printre sistemele denumite FMS.

În fig. 10 se prezintă rezultatul unui studiu privind statistica FMS-urilor din Japonia. Pe axa orizontală este indicat numărul tipurilor de produse realizate cu aceste sisteme, iar pe axa verticală — numărul punctelor de lucru pe muncitor. Această relaţie poate fi interpretată ca rela-

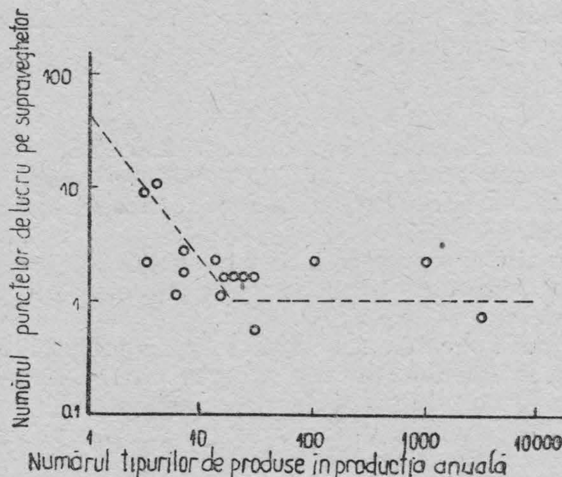


Fig. 10. *Nivelul de automatizare funcţie de cerinţa de flexibilitate.*

ţia între flexibilitatea necesară şi nivelul de automatizare. Este uşor de văzut că flexibilitatea necesară variază foarte mult şi, corespunzător, nivelul automatizării are valori diferite, evidenţiind că există configuraţii diferite de FMS.

Din această figură înţelegem tendinţa generală a automatizării. Este evident că devine greu a automatiza cînd creşte cerinţa de flexibilitate. Este interesant de a exploata curba pe axa verticală. În această figură, ea indică numărul de puncte de lucru (staţii) în care operează un muncitor în cazul producţiei de masă. Valoarea este de 50 sau 60, o valoare rezonabilă, dacă ne gîndim la fabricile cu producţie de masă, ca — de exemplu — în industria de automobile.

Acest fapt conduce la concluzia că producţia de masă este doar un caz special al sistemelor de fabricaţie flexibile. Conceptul FMS este mai general, incluzînd sistemul de fabricaţie în masă unde argumentul, care este numărul felurilor de produse în acest caz, este unitatea.

Dacă acceptăm acest punct de vedere, el va deveni inutil pentru a defini FMS-ul; în schimb, putem spune că un sistem de fabricaţie este caracterizat de un parametru variabil în ce priveşte numărul de tipuri de produse. Totuşi, este necesar a considera aspectele calitative ale tipurilor de produse. De exemplu, tip de produs diferit înseamnă uneori numai dimensiuni diferite ale unor piese cu forme similare. În alte cazuri,

este necesar ca să existe atît piese cilindrice cît și piese prismatice în piesele cerute.

O cale practică pentru a rezolva această problemă este de a aplica clasificarea tehnologiei de grup (GT). Putem schița o curbă de distribuție a formei (spectrul) prin utilizarea acestei clasificări. Un sistem se evaluează, prin asemenea metodă, comparîndu-l pe cel din prezentul Proiect cu alte sisteme de fabricație flexibile.

5.2. Dezvoltarea FMSC

Interesul major în dezvoltarea FMSC este de a proiecta un sistem care să poată fi realizat în serie, pe baza cunoștințelor obținute prin linia pilot, care este denumită *linia test*. Pentru acest scop, trebuie să se facă cercetări mai detaliate asupra alegerii optime și planului complexelor, conforme cu cerințele de fabricație. Unele cercetări fundamentale asupra reprezentării produselor și sistemului de fabricație au fost elaborate iar rezultatele vor fi prezentate în viitorul apropiat.

Dezvoltarea software-ului pentru sistemul actual este rămasă în urmă față de aceea a hardware-ului. Înaintea demarării acestui Proiect, au existat serioase argumente printre cercetători asupra filozofiei de bază a dezvoltării: principalele dezbateri au fost concentrate asupra faptului dacă sistemul trebuie să fie orientat pe hardware sau pe software. Acum este clar că sistemul a fost dezvoltat în mod corect în maniera hardware-orientat, deoarece pe această cale au fost implementate — cu succes — noi concepte de mașini, care par să modifice substanțial metodologiile tradiționale ale fabricației. În etapa actuală se cere că software-ul să fie dezvoltat pentru mai buna utilizare a hardware-ului realizat. Fundamental, sistemul software a fost planificat să aibă o structură isomorfică față de aceea a sistemului hardware, în mod special pentru cazul software-ului de control și al producției. Însă, ca și atunci cînd proiectezi produse, dezvoltarea sistemelor CAD este — mai curînd — la nivelul cercetării de bază, iar dezvoltarea CAD/CAM reală, care este în mod eficient aplicată în acest sistem, pare a fi o problemă de viitor.

Tendința recentă în dezvoltarea CAD/CAM, cum ar fi sistemele CAD inteligente prin utilizarea ingineriei cunoștințelor (IFIP, 1984) și altele, poate fi o soluție bună a acestei probleme.

Una din problemele cele mai importante ale unei asemenea fabrici fără oameni este întreținerea. Cum s-a menționat în precedentele capitole, unele aparate de diagnosticare automată au fost instalate în prezentul sistem. Complexul de control ne va da unele indicații asupra defectării sistemului, cînd sistemul se întrerupe. Este, însă, aproape imposibil să se localizeze în mod exact punctul defecțiunii. Asemenea informație este esențială pentru planificarea operațiilor de reparare. Însă, chiar dacă această informație este disponibilă în mod automat, este imposibil să se repare în mod automat punctul defectat. În sistemul prezent nu există mijloace de a repara în mod automat defecțiunile mecanice; muncitorii trebuie să intre în zona „fără oameni“ pentru a repara în mod manual punctul defect.

Rapoarte recente au relevat că problema întreținerii va deveni una din cele mai importante probleme, practic în toate ramurile industriale (Yoshikawa, 1984).

Așa cum s-a menționat în secțiunea 3.4, unele proiecte au fost implementate pentru dezvoltarea roboților de întreținere, care pot realiza asemenea activități chiar și în centrale nucleare. Merită a discuta posi-

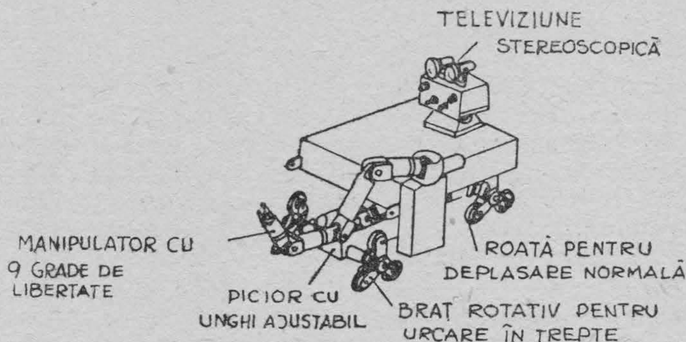


Fig. 11. *Un robot inteligent și mobil.*

bilitatea aplicării acestor roboți la prezentul sistem. Un prototip de robot „inteligent", mobil, a fost dezvoltat în departamentul nostru pentru lucrări de întreţinere (Yoshikawa, 1980); el este reprezentat în fig. 11. Acest prototip este dezvoltat doar pentru experimentări privind lucrările de întreţinere. Este, însă, de dorit și de așteptat ca asemenea roboți de întreținere să fie dezvoltați pentru utilizări practice, din cauza importanței problemei întreținerii în diferite domenii industriale.

Bibliografie

Hatvany, I., M. E., Merchant, K. Rathamil, H. Yoshikawa (1982). Rezultatele unei analize internaţionale asupra fabricării asistate de calculator. *IFAC, Al 8-lea Congres internațional, Preprint CS-124.* Vezi AMC 38 și 39, 1984.

Higashimoto, A., (1983), Cercetarea tehnologiei elementului în sistemul de verificare produs pentru FMC. *Jurnalul societății japoneze a ingineriei de precizie, Vol. 49, nr. 8, pag. 1043.*

Hirano, Y., (1983), Mecanism de asamblare complex în FMC, *ibd. pag. 1036.*

IFIP (1984), WG 5.2 *Conferință de lucru asupra cunoștințelor de engineering și CAD*, Budapesta, septembrie 1984.

Karita, R., (1983), Sistem de mașini complex în FMC, *Jurnalul societății japoneze a ingineriei de precizie, vol. 49, nr. 8, pag. 1028.*

Kitajma, K., H. Yoshikawa (1984), Proiectarea unui sistem ierarhic de mașini HIMADES 1, *Comunicări la CAD 1984*, Butterworth.

Ozani, S., Y. Kato (1983), Sistem total pentru FMC, *Jurnalul societății japoneze a ingineriei de precizie, Vol. 49, nr. 8, pag. 996.*

Tomiyama, T., Yoshikawa (1984), Cerințe și principii pentru un CAD inteligent *IFIP W.G. 5.2, Conferința de lucru asupra cunoștințelor de engineering și CAD.*

Williamson, D.T.N. (1967), Un nou concept de fabricație, *Comunicări la cel de-al 8-lea Sistem 24*, MTDR.

Yoshikawa, H., (1977), Proiectul japonez pentru o fabrică fără oameni, în ed. D.M.C. Pherson, *Prolamat 1976, Realizări în fabricarea asistată de calculator*, North-Holland, pag. 3.

Yoshikawa, H., ș.a. (I: 1980, II: 1981, III: 1982), *Raport asupra cercetărilor și dezvoltării roboticii pentru întreținerea sistemelor*. Asociația de promovare a sistemelor de mașini, Tokyo.

Yoshikawa, H., (1981), Teoria generală de proiectare și un sistem CAD, Editura T. Sata & E. Warman, *Comunicația Om-Mașină în CAD/CAM*, North-Holland.

Yoshikawa, H., (1984), Necesitate și potențial în tehnologia de întreținere, *Comunicări la al 7-lea Congres al Federației Europene a Societăților Naționale de Întreținere*, Veneția.

(continuă la pag. 116)

P.5. JOCURI DE ÎNREPRINDERE. STIMULENTE ȘI CONTROL ÎN ORGANIZAȚIILE ECONOMICE

Ray Radner

AT8T Bell Laboratories 600 Mountain Avenue Murray Hill, NJ 07974, S.U.A.

Articolul trece în revistă unele teorii ale descentralizării economice, începînd cu teoria echipelor. Teoria echipei este legată de utilizarea eficientă a informației într-o organizație în care diferiți factori de decizie nu au, în mod obligatoriu, informații identice. Eficiența este evaluată prin conceptele de *scop organizațional total* sau *funcție obiectivă*, fără a se acorda o atenție deosebită stimulentelor particulare ale factorilor de decizie individuali. După o introducere în teoria echipelor, se continuă cu trecerea în revistă a diferitelor probleme stimulatoare, incluzînd „denaturarea", riscul moral și libera concurență. Aceste probleme și remediile propuse sînt discutate în mod detaliat într-o serie de exemple de relații cu alocare de resurse, activități superioare, relații întreprinzător — reprezentant și de asociere și micșorarea riscului moral în relațiile pe termen lung. În încheiere sînt făcute unele observații asupra implicațiilor posibile a acestor teorii pentru proiectarea organizațiilor economice și pentru explicarea comportării organizaționale.

1. INTRODUCERE

Pentru mulți economiști, cuvîntul *„descentralizare"* înseamnă *„piețe"* și tot pentru mulți economiști piețele competitive reprezintă o formă superioară de descentralizare economică.

În această expunere se adoptă un punct de vedere mai larg decît acesta și diferit orientat. Va fi mai larg în sensul că conceptul descentralizării ce va fi explorat este aplicabil celor mai complexe organizații economice. Orientarea va fi diferită pentru că motivația în studierea teoriilor descentralizării este o dorință de a înțelege relațiile economice în interiorul marilor organizații moderne cum ar fi corporațiile.

Cu toate acestea, unele metode teoretice ce vor fi descrise sînt aplicabile, de asemenea, relațiilor de piață.

Prin organizație *descentralizată* se va înțelege aici o organizație cu mai mult decît un singur factor de decizie, în care diferiții factori de decizie sînt responsabili pentru diferite hotărîri variabile, iar acele hotărîri pe baza unor informații diferite și în care efectul asupra organizării depinde organic de mai multe decizii și de unele variabile stochastice de mediu. Această destul de generală și abstractă definiție va fi mai bine înțeleasă de cititor cînd se va întîlni cu modelele speciale care vor fi expuse în acest articol.

Una din temele care vor fi ilustrate în acest articol este aceea că descentralizarea informației într-o organizație (de ex. incompleta acumulare de informație), cînd este combinată cu un dezacord al intereselor factorilor de decizie, conduce la hotărîri ineficiente. Această ineficiență merge dincolo de ceea ce se datorează informației incomplete în sine.

Cu alte cuvinte, combinarea unei incomplete acumulări de informații cu dezacordul intereselor conduce la o hotărîre care este mai puțin efi-

cientă decît cea care ar putea fi teoretic obținută dacă un calculator ar fi programat să dea decizia cu aceeași structură incompletă a informației.

Fiecare organizație economică a construit în interiorul său unele sisteme sau reguli prin care compensarea membrilor depinde de activitățile lor și de efectele acestor activități.

Desigur, membrii organizației vor evalua nu numai compensarea lor economică, dar de asemenea, natura muncii lor, acțiunile pe care ei și alții le desfășoară etc. Situația rezultată poate fi analizată cu instrumentele teoriei jocurilor de grup, ceea ce se va face în continuare. Mai exact, se va presupune că se poate prevedea comportarea organizației ca fiind un echilibru necooperativ al jocului corespunzător. (Se va defini acest concept de echilibru în Secțiunea 4 și va fi ilustrat în secțiunile următoare). Unul din punctele care trebuiesc subliniate, cu toate acestea, este că echilibrele care sînt necooperative în sensul tehnic pot manifesta comportări care ar putea fi numite „cooperative" în limbajul curent.

Regulile organizațiilor nu sînt imuabile, desigur și unul din scopurile normativelor economice este de a înțelege ce fel de organizare economică este cea mai potrivită pentru a promova eficiența și echitatea. Teoriile descentralizării evidențiază două tipuri de diversități printre membrii unei organizații care pot reduce eficiența sa:

1. O diversitate a informației, care la rîndul ei poate fi reprodusă de costurile observației, comunicației și calculului.

2. O diversitate a intereselor sau scopurilor.

Din perspectiva analitică a prezentului articol, problema organizării optimale poate fi rezumată ca:

Alegerea unei scheme organizaționale pentru care echilibrele jocului corespunzător sînt cît mai bune cu putință, identificînd constrîngerile impuse de diversitatea informațiilor și intereselor între membrii organizației.

Această abordare a schemei organizaționale a fost evidențiată de Hurwicz (1972).

În acest articol vor fi schițate unele dezvoltări recente în teoria descentralizării, în relație primară cu problemele de stimulare ale „riscului moral" (Secțiunea 6) și „libera concurență" (Secțiunea 7). Se va indica modul cum pot fi remediate ineficiențele acestor fenomene, cel puțin în parte, prin exploatarea relațiilor pe termen lung, cu condiția ca membrii organizației să nu aibă vederi prea înguste (Secțiunea 8). Va fi, de asemenea, atinsă o altă problemă de stimulare numită „denaturarea" și indicată o clasă de remedii (Secțiunea 5). Cu toate acestea, discuția asupra acestui ultim punct va fi relativ scurtă, datorită existenței unei tratări extinse a acestei teme (Vezi Groves and Ledyard — 1983, Green and Laffont — 1979 și Myerson — 1983).

Cele mai multe din referințele din literatura de specialitate sînt incluse în notele bibliografice de la sfîrșitul articolului. În particular, cititorul va trebui să consulte referințele corespunzătoare pentru cele mai generale expresii matematice și verificări ale teoremelor din prezentul articol.

2. Un exemplu tipic de descentralizare într-o firmă

Se va începe prin prezentarea unui exemplu tipic al alocării resurselor între filialele unei firme. Acest exemplu va ilustra unele din rezultatele care au fost formulate în teoriile descentralizării. Se va continua apoi cu discutarea rezultatelor individuale, în mod detaliat, cu ajutorul exemplelor mai simple derivate din acesta.

Se consideră o firmă constituită dintr-un sediu central și un număr de filiale. Rolul sediului central sau al „centrului" cum va fi numit în continuare este să aloce filialelor, mijloacele dirijate centralizat. De exemplu, aceste mijloace pot fi fonduri de investiție care pentru scopurile urmărite în acest exemplu vor fi numite „capitaluri". (În fapt pot exista mai multe mijloace dirijate centralizat, de exemplu spațiul, timp de utilizare a calculatoarelor etc., dar pentru simplificare se va presupune existența unuia singur.)

Fiecare filială produce un venit net care poate fi măsurat în aceleași unități, de exemplu dolari. Venitul net al fiecărei filiale depinde de trei „factori".

1. de activitățile directorului filialei, care va fi denumită „activitatea filialei";
2. de cantitatea de capital ce i s-a alocat de către centru și
3. de variabile aleatorii exterioare care vor fi denumite „factori externi" (restricții) ai filialei.

Fie A_i acțiunea directorului filialei i $(i=1,\dots N)$ și A_o acțiunea centrului. Acțiunea centrului este de fapt o alocație

$$A_o=(K_1,\dots K_N)$$

a resursei („capitalului") către filiale, unde K_i este cantitatea de capital alocată filialei i. Dacă totalul capitalului disponibil este K, atunci alocația trebuie să satisfacă inegalitatea:

$$K_1+\dots+K_N\leqslant K$$
$$K_i\geqslant 0,\ i=1,\dots N \tag{2.1}$$

Fie X restricția firmei; ea include între altele o specificare a capitalului total, K. Venitul total C al firmei depinde de acțiunile directorilor filialelor, de alocațiile de capital și de factorii externi (restricții):

$$C=G(A_1, K_1\dots A_N, K_N, X) \tag{2.2}$$

În particular, se consideră cazul special în care venitul total este suma veniturilor filialelor C_i și restricțiile firmei X, pot fi descompuse în restricțiile filialelor, X_i $(i=1\dots N)$ și restricția centrului X_o, care este egal cu capitalul total $(X_o=K)$. Se presupune că venitul firmei i $(=1\dots N)$ este:

$$C_i=G_i(A_i, K_i, X_i) \tag{2.3}$$

și venitul total este:

$$C=C_1+\dots+C_N \tag{2.4}$$

Înainte de stabilirea unei acțiuni, fiecare director $i\,(=0\ldots N)$ primește niște informații Y_i; acțiunea A_i este atunci aleasă în conformitate cu funcția de decizie D_i

$$A_i = D_i(Y_i) \qquad (2.5)$$

Dacă $X=(X_o\ldots X_N)$ și $Y=(Y_o\ldots Y_N)$ sînt variabile aleatorii cu o distribuție de probabilitate dată, atunci pentru oricare din funcțiile de decizie date $D_o\ldots D_N$, acțiunile directorului $A_o\ldots A_N$ și restricția X va exista o distribuție de probabilitate conexată, care va induce la rîndul său o distribuție de probabilitate a venitului filialei și a venitului total, în conformitate cu (2.4) și

$$C_i = G_i[D_i(Y), D_o(Y_o), X_i] \qquad (2.6)$$

(Se reamintește că $K_i=D_o(Y_o)$. Pentru a completa modelul unei astfel de organizări, se va arăta cum este generată structura informațională $Y=(Y_o\ldots Y_N)$ și cum sînt determinate funcțiile de decizie $D_o\ldots D_N$.

3. Teoria echipelor

Teoria echipelor, un prim stadiu în cadrul teoriei organizării, este interesată în utilizarea eficientă a informației într-o organizație informațional descentralizată. Eficiența este evaluată în termenii scopului organizațional total.

Accentul se pune pe: (1) distribuția incompletă a informației între diferiți factori de decizie (descentralizare informațională), (2) caracterizarea funcțiilor de decizie optimale, dată de descentralizarea informațională și (3) compararea structurilor informației (descentralizate) alternative, presupunînd că fiecare va fi utilizată eficient.

În cadrul acestei teorii, nu este acordată nici o atenție stimulentelor particulare, factorilor de decizie individuali, cum ar fi directorii filialelor și centrul, în exemplul din Secțiunea 2. Acești factori de decizie ar putea fi foarte bine calculatoarele; preocuparea teoreticianului este în acest caz să programeze aceste calculatoare în vederea utilizării eficiente a informației disponible.

Pentru a ilustra conceptul de echipă trebuie revenit la exemplul alocării resurselor din Secțiunea 2. În acel exemplu, incertitudinea în legătură cu restricția firmei este reprezentată de o distribuție de probabilități conexate ale variabilelor restricției. O structură informațională dată este generată de unele procese în care diferiți membri ai echipei (centru, directorii filialei) fac — posibil incomplet și imperfect — observații asupra restricțiilor lor, urmate de procese de comunicație mai mult sau mai puțin incomplete. De aici, informația rezultată poate fi, de asemenea, reprezentată de variabile aleatorii a căror distribuție de probabilitate conexată cu restricția este determinată de structura informației date, cum ar fi un proces dat de observație și comunicare. Dacă se introduce această informație în strategiile membrilor echipei, produsul rezultat va fi de

asemenea o variabilă aleatorie. De notat că natura stohastică a produsului derivă din natura stohastică a restricției în două moduri: (1) direct, prin influența restricției asupra funcțiilor de producție ale filialelor și (2) indirect, pe calea care conduce de la mediu la informație și apoi la acțiuni.

Se presupune că funcția obiectivă a echipei este obținerea venitului total sperat al filialei. Fiind date distribuția conexată a restricțiilor și variabilele informației, problema majoră a echipei este să găsească acele strategii pentru centru și filiale care să maximizeze venitul sperat al echipei. Acestea se numesc *strategii optimale ale echipei pentru o structură dată a informației.*

În principiu, alegerea structurii informației în sine ar putea fi o parte din structura organizatorică a echipei. Activitățile care generează o structură informațională presupun cheltuieli, astfel încît diferite structuri pot avea costuri diferite. O structură informațională este mai bună decît alta dacă la un cost estimat net ea produce un venit sperat mai mare. Din păcate, foarte puține lucrări teoretice se ocupă de modele explicite ale costurilor informației, astfel că în cele ce urmează costurile informației se vor considera implicite. Astfel, structurile informaționale vor fi comparate pe baza cheltuielilor (brute) pe care le produc în timpul utilizării, se înțelege, cu strategiile optimale ale echipelor corespunzătoare.

În continuarea exemplului din Secțiunea 2 se vor descrie patru structuri informaționale diferite.

În prima, „Fără informare“ [FI], fiecare director studiază numai propria sa restricție:

$$Y_o = X_o = K$$
$$Y_i = X;\ i \geqslant 1 \qquad (3.1)$$

În a doua, „Schimb complet cu centrul“ [SCC] fiecare filială schimbă informațiile sale cu centrul dar nu și cu alte filiale:

$$Y_o = (K,\ X_1 \ldots X_N)$$
$$Y_i = (X_i,\ K);\ i \geqslant 1. \qquad (3.2)$$

În a treia, „Informare completă“ [IC], toate informațiile sînt schimbate:

$$Y_i = (K,\ X_i \ldots X_N)$$
$$i = 0 \ldots N. \qquad (3.3)$$

Este clar că creșterea informării este în ordinea (FI), (SCC), (IC).

Pentru orice structură informațională dată, funcțiile de decizie corespunzătoare echipei optime sînt acelea care maximizează venitul total sperat al firmei. S-ar putea arăta că acest venit optimal sperat va fi (în general) mai mare pentru (SCC) decît pentru (FI) și mai mare pentru (IC) decît pentru (SCC). (Există cazuri speciale în care aceste inegalități devin egalități.)

A patra structură informațională „Studiul complet“ (SC) este cel mai interesant dar mai complicat. Fiecare director începe prin observarea propriilor sale restricții. Centrul comunică un mesaj, M_o, tuturor directorilor filialelor (un „preț“) care este o funcție a capitalului total furnizat:

$$M_o=\mu_o(K). \quad (3.4)$$

Fiecare conducător de filială comunică apoi un mesaj, M_i, centrului, („cererea“ sa de capital) care este o funcție a mesajului „preț“ și a propriei sale restricții:

$$M_i=\mu_i(M_o, X), i\geqslant 1 \quad (3.5)$$

Structura informațională rezultată este:

$$\begin{aligned} Y_0&=(M_1 \ldots M_N, K) \\ Y_i&=(M_0, X_i), i\geqslant 1 \end{aligned} \quad (3.6)$$

Mesajele filialelor sînt calculate într-un mod particular.

Fie $(\overline{A}_i, \overline{K}_i)$ „profitul imagine“ maximizat al filialei i:

$$(\overline{A}_i, \overline{K}_i) \text{ maximizează } G_i(A_i, K_i, X_i)-M_0K_i \quad (3.7)$$

atunci:

$$M_i=\overline{K}_i. \quad (3.8)$$

Este clar că (SC) este situat între (FI) și (SCC) din punctul de vedere al capacității de informare. În mod corespunzător, venitul total maxim sperat pentru structura „SC“ este situat între cel corespunzător structurii (FI) și structurii (SCC).

Este demn de atenție faptul că pentru funcțiile de decizie ale echipei optimale corespunzătoare structurii SC, capitalul K_i, alocat de fapt filialei „i“ nu va fi de regulă egal cererii $\overline{K}_i$ și nici A_i nu va fi de regulă egal cu $\overline{A}_i$. Aceasta este urmarea faptului că cererea totală nu este de regulă egală cu capitalul total livrat. Structura informațională (SC) poate fi interpretată ca primă fază într-o procedură interactivă de tatonare pentru găsirea unui „echilibru competitiv“ pe piața internă pentru resursa alocată de centru, capitalul.

În cadrul acestui material nu este spațiu suficient pentru a discuta proprietățile și evaluarea strategiilor optimale pentru diferitele structuri informaționale descrise mai sus, dar pot fi menționate pe scurt unele rezultate ale comparării acestor structuri. Pentru aceste comparații atenția va fi limitată la o subfamilie a structurilor (SC) pentru care interpretarea dată „pieții“ este în mod particular, clară.

Pentru orice mesaj de preț dat și alocare a capitalului unei filiale, fie volumul capitalului produsul prețului și cantității alocate și fie profitul filialei, diferența între venitul său curent și volumul capitalului alocat ei. Se definește funcția „cerere“ a filialei astfel: directorul filialei determină o activitate a filialei și o cantitate a capitalului care maximizează profitul condițional sperat al filialei, dat de restricțiile filialei și de mesajul de preț primit de la centru; această cantitate de capital este comunicată centrului ca cererea filialei. De notat că acest calcul făcut

de directorul filialei impune cunoașterea funcției de preț utilizată de centru și nu doar mesajul preț, în particular, primit.

Se presupune că restricțiile membrilor echipei sînt statistic independente și că funcția preț este aleasă astfel încît să optimizeze structura (SC) în această categorie. Radner (1972 b) a arătat că pentru un caz particular al funcțiilor pătratice a venitului filialei se poate obține cu structura informațională SC un venit al echipei, la fel de ridicat ca și cu structura SCC. (Groves și Radner — 1972 — au generalizat acest rezultat pentru cazul în care este alocată mai mult decît o singură resursă). Deși în cazul general (nepătratic) structura SCC va fi mai bună decît cea mai bună structură SC, Groves (1969) a arătat că structura SC poate fi aproximativ la fel de bună ca structura SCC dacă funcțiile de producție sînt aplatizate și distribuțiile restricțiilor nu sînt prea dispersate.

Două aspecte ale structurilor SC (optimale) sînt interesante. Primul, capitalul total cerut de filiale nu va fi (decît accidental) egal cu cel pus la dispoziție. Procesul care urmează în condițiile unei structuri SC este analog unei prime iterații într-un proces de tatonare, dar desigur, echilibrul nu este obținut printr-o singură iterație. Cu toate acestea, în cazurile speciale la care s-a făcut referire mai sus, structura SC este în mod remarcabil eficientă.

Al doilea aspect, constă în aceea că în structura SC activitatea optimală a filialei va diferi de regulă de activitatea care maximizează profitul sperat calculat al filialei.

Cu toate că o explicație intuitivă a acestui fenomen nu este evidentă, ea este în mod clar în raport cu faptul că „cererea“ totală și livrarea nu sînt în echilibru.

Un al doilea set de rezultate privesc eficiența structurii SCC. Arrow și Radner (1979) au arătat că, dacă restricțiile membrilor echipei sînt statistic independente și dacă unele proprietăți de regularitate sînt satisfăcute (inclusiv concavitatea funcțiilor de producție și simetria filialelor), atunci venitul (optimal) așteptat pentru fiecare filială se va apropia de un maxim posibil în cadrul structurii IC, pe măsură ce numărul filialelor crește. În termeni aproximativi, într-o echipă „numeroasă“ cu restricții statistic independente, structura SCC este aproape la fel de bună ca structura IC, dacă nu chiar mai eficientă. (Rezultatele mai puțin satisfăcătoare în această idee au fost demonstrate mai înainte pentru cazul pătratic, de Radner (1972 b) și de Groves și Radner (1972)). Rezultatele obținute de Arrow și Radner au fost generalizate recent și îmbunătățite în mod vizibil de Groves și Hart (1982) pentru a demonstra eficiența asimptotică a structurilor informaționale de tip SC, care sînt net mai puțin bune decît structurile SCC.

4. Probleme de stimulare

Teoria echipelor, așa cum a fost descrisă în secțiunea precedentă, implică utilizarea eficientă a informației într-o organizație descentralizată dar dintr-un punct de vedere limitat. „Eficiența“ este evaluată în termenii unor scopuri organizaționale totale și principalele probleme puse

de această teorie sînt: (1) caracterizarea strategiilor optimale a echipei, pentru o structură dată a informaţiei (descentralizată) şi (2) compararea structurilor informaţionale alternative, presupunînd că fiecare va fi eficient utilizată.

Cum ar trebui extinsă teoria pentru a ţine seama de scopurile şi obiectivele personale ale factorilor de decizie individuali? Iacob Morschak şi-a dat seama de această problemă cînd, primul, a început să lucreze la teoria echipei şi a afirmat că, cadrul adecvat era acela al teoriei jocurilor, de curînd apărută. Cu toate acestea, în timpul acela (cu mai mult de 20 ani în urmă) teoria jocurilor nu punea la dispoziţie un ghid clar asupra „conceptului soluţie" care ar fi adecvat pentru studiul stimulentelor (unii critici argumentează că nu ar fi totuşi aşa!) şi explorarea celor mai promiţătoare concepte erau încă în stare incipientă. Teoria echipelor a fost astfel primul punct pe o agendă de cercetări pe termen lung şi acum se revine la ea cu progrese recente.

În această acţiune se va trece în revistă, într-un mod abstract dar informativ, problemele stimulentelor în cadrul echipelor, iar în secţiunile următoare vor fi ilustrate unele din aceste idei cu ajutorul mai multor exemple.

Care sînt ţelurile personale ale factorilor de răspundere într-o organizaţie? A pune întrebarea în termeni economici, din ce surse, un factor de decizie obţine un „cîştig" (sau „pierdere")? Pentru scopurile propuse în această secţiune este util să se distingă două surse ale cîştigului. Prima, cîştigul unui factor de decizie poate fi afectat direct de activităţile sale personale şi de cele ale altora: de ex. cît de dificilă şi îndelungată este munca sa, condiţiile sale de muncă etc. A doua, cîştigul, va fi afectat de compensaţia pe care o primeşte de la organizaţie. Această compensaţie va depinde pe de altă parte, de activităţile diferiţilor factori de decizie, fie direct (de ex. activităţile pot fi verificate pentru a stabili dacă sînt corecte, activităţile incorecte putînd conduce la penalizări), fie indirect (de ex. prin participare la beneficii, gratificaţii în funcţie de mărimea venitului organizaţiei).

În continuare se va reveni la exemplul din Secţiunea 2.

Se presupune că activităţile unui director de filială afectează direct cîştigul său: spre exemplu, el preferă activităţile care cer un efort mai mic, celor care cer un efort mai mare. Dacă strategiile optimale ale echipei solicită directorul pentru mai mult decît efortul minim, atunci va fi necesar un „mecanism de compensare" (MC) pentru a-l determina să depună un efort suplimentar.

În cele ce urmează, sînt date trei exemple de mecanisme de compensare pentru un director de filială:

— (MC1) Activităţile directorului sînt monitorizate (sau verificate din exterior) şi el primeşte o retribuţie fixă dacă a folosit corect funcţia de decizie şi nimic în caz contrar.

— (MC2) Directorul primeşte o parte fixă din venitul filialei sale plus o plată fixă (care poate fi pozitivă sau negativă).

— (MC3) Directorul primeşte o parte fixă din totalul venitului echipei plus o plată fixă (care poate fi pozitivă sau negativă).

Fiecare din aceste mecanisme de compensare are problemele sale. În vederea aplicării mecanismului MC1 conducerea externă trebuie să stabilească nu numai activitatea ce trebuie desfășurată dar, de asemenea, și informația pe care este bazată. În măsura în care descentralizarea este cauza costurilor distribuirii și prelucrării informației, o astfel de monitorizare detaliată va fi costisitoare sau impracticabilă. Astfel, MC1 poate oferi directorului stimulente reale dar la un cost ridicat sau chiar prohibitiv.

Cerințele informaționale pentru mecanismele MC2 și MC3 sînt mai modeste dar fiecare are probleme de stimulare. Înainte de a discuta aceste probleme în detaliu, trebuie să fie mai bine clarificată noțiunea de „eficiență" care urmează a fi folosită.

Un punct de vedere ar putea fi acela al proprietarului întreprinderii care, pentru scopurile propuse în acest exemplu, va fi indentificat cu centrul. Pentru proprietar orice compensație cuvenită directorilor filialelor este o cheltuială în afacerile curente și urmează a fi scăzută din venitul întreprinderii. În acest caz „eficiența" este măsurată de profitul net sperat al întreprinderii (venitul mai puțin compensațiile de conducere). Acesta se va numi „*modelul generalizat întreprinzător-reprezentant*" (proprietarul este „întreprinzătorul" și directorii filialelor sînt „reprezentanții").

O altă noțiune a eficienței ar putea fi adoptată în cazul în care directorii au fost ei înșiși proprietari asociați ai întreprinderii (în acest caz, centrul ar trebui să fie unul din directori). Aceasta se va numi „*modelul asocierii*". În acest caz ar fi natural să se adopte noțiunea de „eficiență uniformă": o combinație a funcțiilor mesaj și a strategiilor este uniform eficientă, dacă nici o altă combinație nu ar face cel puțin un director absolut mai bogat și nici unul mai sărac. Cu această abordare s-ar putea lua în considerare cîștigul sperat al fiecărui director, care ar reflecta cîștigul direct datorat propriilor sale activități ca și compensației sale.

Chiar în cadrul modelului generalizat „întreprinzător-reprezentant", s-ar putea lua în considerare cîștigurile sperate a directorilor, precum și acelea ale proprietarului, în care caz, s-ar putea din nou ajunge la norma eficienței uniforme.

În continuare, se vor considera problemele de stimulare care sînt inerente în mecanismul de compensare MC2, în contextul modelului generalizat „întreprinzător — reprezentant", adică cu un „proprietar — centru". Întîi, în afara cazului că directorul filialei este neutru la risc, va fi în mod obișnuit imposibil să se găsească un mecanism de compensație pentru forma MC2 convingător pentru cel ce ia decizii cu eficiență uniformă. De exemplu, dacă fracțiunea venitului filialei pe care o primește directorul este mai mică decît 1, atunci nivelul său de efort va fi inferior nivelului la care pierderea marginală a efortului este situată chiar în afara valorii „sociale" marginale a creșterii corespunzătoare în venitul sperat al filialei. Această problemă de stimulare ilustrează mai generala problemă a „riscului moral". Un caz de excepție este acela în care directorul filialei este neutru la risc; dacă aceasta și alte condiții sînt satisfăcute (se va reveni la aceasta mai tîrziu), atunci riscul moral poate fi eliminat prin bunurile pe care fiecare director de filială le cumpără

sau în cazul în care închiriază filiala de la proprietar, atunci revenindu-i întregul venit al filialei (Acesta este un caz special al mecanismului MC2, în care partea directorului din venitul filialei este 1 și el primește o plată fixă *negativă* de la proprietar).

O a doua problemă de stimulare în mecanismul MC2 se referă la mesajele directorului filialei către centru. Pentru a fi mai clar, se consideră o structură informațională SC (vezi secțiunea 3).

Pentru orice funcție dată pe care directorul filialei se presupune că o utilizează pentru calculul cererii sale de capital, va fi stimulat să exagereze cererea sa pentru a putea produce un venit mai mare (și astfel să crească și compensația sa). Această situație ilustrează o problemă mai generală a *„denaturării“*. În exemplul de față cineva ar putea fi ispitit să remedieze această problemă modificînd mecanismul de compensare astfel încît directorul filialei să justifice capitalul cerut (sau care îi este alocat). Se poate arăta că o modificare a acestui lucru va fi efectivă în condiții particulare: chiar în cazul acesta, nu va fi o situație obișnuită ca justificarea capitalului de către directorul filialei să elimine stimulentul pentru declarații false.

Denaturarea descrisă mai sus poate avantaja un director de filială dar cu pierderi din venitul altor filiale. De aceea, mecanismul MC3 ar putea fi considerat un remediu al acestei situații pentru că directorul fiecărei filiale va avea interesul să crească venitul tuturor filialelor și nu numai a celei pe care o conduce. Cu toate acestea, o orientare către MC3 ar prezenta o problemă suplimentară de stimulare: fiecare director ar fi acum un „liber concurent“ al eforturilor celorlalți directori.

Într-un anumit sens, aceasta ar exacerba problema riscului moral descris mai înainte.

Pentru a concluziona discuția precedentă, se poate spune că pentru orice structură informațională dată și mecanism de compensație, acțiunile și mesajele pot genera aspecte exterioare pentru alți directori, aspecte care pot fi reflectate necorespunzător în stimulentele primului director și conduce la scăderea eficienței întreprinderii.

În mod special, au fost identificate trei probleme generale de stimulare:

1 — riscul moral;
2 — concurența;
3 — denaturarea.

Pînă în acest punct nu a fost precizat modelul exact utilizat pentru a prevedea comportarea directorului în această situație.

În cele ce urmează se va adopta cadrul teoriei jocurilor necooperative și se va utiliza conceptul „echilibrului necooperativ“ pentru a prevedea comportarea participanților la joc (directorii). Formal, un echilibru necooperativ este o combinație a strategiilor, una pentru fiecare jucător, astfel încît nici un jucător nu-și poate crește cîștigul personal prin schimbarea unilaterală a propriei strategii[1]. (În cele ce urmează, termenul

[1] Acesta este uneori denumit echilibrul Nash sau Cournot-Nash.

„echilibru“ va fi înțeles ca „echilibru necooperativ“ dacă nu se precizează altfel). O schemă particulară a unei organizații (structură informațională, mecanism de compensații etc.) va determina un anumit set de reguli a jocului. *Problema organizării optime poate fi atunci privită ca o alegere a schemei pentru care echilibrul (sau echilibrele) jocului corespunzător sînt cît mai eficiente.*

În continuarea acestui material vor fi ilustrate problemele denaturării, a riscului moral și a concurenței în cîteva jocuri teoretice a modelelor întreprinzător — reprezentanți și de asociere. Se va sublinia că aceste relații simple sînt interesante, nu numai izolat dar și pentru că ele sînt fixate în structura mai largă a organizațiilor economice complexe obișnuite.

5. „Denaturarea“: stimulente și surse publice

Pentru a ilustra mai detaliat problema denaturării se va discuta în continuare o variație a alocării de resurse exemplificată în Secțiunea 2.

Acest exemplu va fi, de asemenea, utilizat pentru a ilustra o clasă de mecanisme de compensație care, în unele condiții, pot fi utilizate pentru remedierea ineficienței care poate fi cauzată de obicei de denaturare, ca stimulent.

Se presupune că resursa dată de centru este un „supliment“ sau în terminologia economică un „bun public local“. Pentru a fixa ideea, se va denumi resursa „cercetare“, iar centrul care face cercetarea, „sediul central“ al firmei. Obiectivul sediilor centrale este de a face cercetarea, pe cheltuiala lor și de a produce rezultate pe care să le pună la dispoziția tuturor filialelor. Un program particular de cercetare va produce beneficii pentru fiecare filială, în sensul că va crește venitul net pe care filiala îl poate produce. Întrucît stabilirea mărimii bugetului de cercetare trebuie făcută înainte de a cunoaște beneficiile reale, măsura adecvată a beneficiului este „beneficiul sperat“; acesta se va numi pe scurt „beneficiu“. Sediile centrale vor trebui să adopte un program de cercetări care să maximizeze diferența dintre beneficiile totale (sperate) și costuri.

Costul unei astfel de activități este în mod obișnuit distribuit între filiale, astfel că pentru scopuri contabile venitul net al filialei trebuie redus la costul cercetării care îi este alocat. Astfel, din punctul de vedere al filialei efectul adoptării de către sediul central a unui program particular de cercetări este 1) să crească venitul său cu cantitatea de beneficiu produsă de cercetare și 2) să descrească venitul cu volumul de cheltuieli necesitate de cercetare.

Diferența dintre aceste două efecte se va numi beneficiul net al filialei. Dacă performanța directorului filialei este măsurată de venitul său net, atunci el va dori ca sediile centrale să adopte programul de cercetări care îi produce cel mai mare beneficiu net.

Problema „denaturării“ se pune cînd fiecare director de filială are informații mai bune asupra beneficiului (sperat brut) al filialei sale produs de un program special de cercetări, decît sediile centrale. În acest caz, sediile centrale vor încerca să îmbunătățească deciziile lor cerînd informații suplimentare de la filiale. Pe de altă parte, o formulă specială de alocare de cheltuieli ar putea rezulta în situația în care un program de cercetare dat va produce deficite pentru unul sau mai mulți directori de filiale dar un beneficiu net pozitiv pentru firmă în totalitate (sau invers). Acest conflict de interese ar putea stimula directorii să declare fals beneficiile adevărate aduse de cercetare.

Fie P_j beneficiul real pentru filiala j, al creșterii propuse R, în bugetul centralizat de cercetare, fie M_j beneficiul său impus adică mesajele către sediul central și fie T_j cheltuielile calculate (impozite) alocate filialei j. Filialele trimit mesajele lor simultan și sediul central adoptă creșterea dacă

$$\Sigma M > R. \tag{5.1}$$

Prima variantă a Impozitului Groves — Loeb este dată de

$$T_j = \begin{cases} R - \sum_{k \neq j} M_k & \text{dacă este adoptată creșterea} \\ 0 & \text{dacă nu este adoptată} \end{cases} \tag{5.2}$$

Beneficiul net al filialei j care constituie creșterea este $P_j — T_j$. Deci, filiala j are nevoie de creșterea adoptată dacă

$$P_j > R - \sum_{k \neq j} M_k \tag{5.3}$$

Pe de altă parte, sediul central va adopta creșterea dacă

$$M_j > R - \sum_{\neq kj} M_k \tag{5.4}$$

Comparînd relațiile (5.3) și (5.4) se observă că, oricare mesaje sînt trimise de filiale altele decît j, filiala j va determina sediul central să ia o decizie care este în interesul filialei j făcînd $M_j = P_j$, adică prin declararea adevărului. Deci declararea adevărului este o strategie dominantă de echilibru.

Este simplu de verificat că în conformitate cu această primă variantă taxele totale nu pot depăși valoarea R, adică veniturile nete ale sediului central nu sînt pozitive. A doua metodă Groves-Loeb garantează că veniturile nete ale sediului central nu sînt negative. Fie (C_j) orice numere pozitive a căror sumă este R și se definește:

$$V_j = \sum_{k \neq j} (M_k - C_k)$$

Atunci a doua metodă este dată de:

$$T'_j = \max.\ (0,\ V_j) + \begin{cases} T_j, & \text{dacă se adoptă} \\ 0 & \text{dacă nu se adoptă} \end{cases} \tag{5.5}$$

De notat că T diferă de T_j printr-o cantitate care nu depinde de M_j. Deci, metoda a doua furnizează aceleași stimulente ca și prima. Acum se poate rescrie (5.5) astfel:

$$T'_j = \begin{cases} C_j + \max. (0, -V_j) & \text{dacă se adoptă} \\ \max. (0, V_j) & \text{dacă nu se adoptă} \end{cases} \quad (5.6)$$

Această ultimă formă arată clar că $T'_j \geqslant C_j$ dacă creșterea este adoptată și $T'_j \geqslant 0$ dacă nu este adoptată, astfel că veniturile nete ale sediilor centrale sînt pozitive. (Desigur, conform metodei a doua o filială poate plăti o taxă pozitivă chiar dacă creșterea nu este adoptată).

6. Riscul moral: modelul „întreprinzător — reprezentant"

Modelul „întreprinzător — reprezentant" este cel mai simplu model teoretic în care se poate studia fenomenul riscului moral. De fapt, el este un caz special al modelului lui Groves al „stimulentelor în echipe" în care există un organizator, *întreprinzătorul* și un factor de decizie, *reprezentantul*. În acest model rezultatul este dependent de activitatea reprezentantului și de o restricție stohastică, dar întreprinzătorul nu poate dirija complet informațiile și activitatea reprezentantului și nici restricția. Întreprinzătorul poate determina efectul, cu toate acestea, și în cea mai simplă formă a modelului, întreprinzător-reprezentant — una va fi studiată în continuare — acesta este singurul lucru care poate fi dirijat. Astfel, în cel mai simplu caz, întreprinzătorul poate face astfel încît compensația reprezentantului să depindă în primul rînd de rezultate. Mai general, compensația poate depinde de orice altceva pe care întreprinzătorul îl poate examina, de exemplu o informație incompletă despre informarea, activitatea sau restricțiile reprezentantului.

În această secțiune va fi utilizat un exemplu foarte simplu al modelului întreprinzător-reprezentant pentru a ilustra cum riscul moral poate conduce la ineficiență. Se presupune că rezultatul (stohastic) al întreprinderii este „succes" sau „faliment" și că probabilitatea de succes depinde de activitatea reprezentantului. În cazul succesului întreprinzătorul cîștigă o cantitate de bani (de exemplu o mie de dolari) dar în cazul falimentului nu cîștigă nimic. Întreprinzătorul va compensa reprezentantul corespunzător, acordîndu-i o compensație w_1 pentru succes și o compensație w_0 pentru faliment. (În principiu, compensația poate fi negativă cu toate că reglementări internaționale ar putea prevedea demiterea).

Cîștigul întreprinzătorului este presupus a fi egal cu diferența dintre rezultat și compensația pe care o plătește reprezentantului. (În acest fel întreprinzătorul este neutru față de risc.) Cîștigul reprezentantului este presupus a depinde atît de activitatea cît și de compensația sa. (Reprezentantul poate fi neutru sau refractar față de risc.)

Se va reprezenta această situație ca un joc în două faze. Acțiunea reprezentantului este un număr real pozitiv, A, și venitul rezultat este:

$$C = G(A, X),$$

unde X este o variabilă aleatorie. În acest exemplu, X este distribuit uniform în intervalul 0...1 și

$$G(A, X)=\begin{cases} 1 \text{ dacă } A\geqslant X \\ 0 \text{ dacă } A<X \end{cases}$$

Se poate interpreta C ca succes sau faliment, X ca dificultate a sarcinii reprezentantului și A, efortul său. Din definirea lui G rezultă:

$$\text{Prob } (C=1)=\text{min. } (A, 1)$$

Reprezentantul nu are nici o informație despre X (alta decît distribuția ei) cînd alege acțiunea sa.

Compensația reprezentantului depinde de venitul C conform cu:

$$R(C)=\begin{cases} w_0 & \text{dacă } C=0 \\ w_1 & \text{dacă } C=1 \end{cases}$$

Cîștigul reprezentantului este:

$$U_1=P[R(C)]-Q(A)$$

Unde P și Q sînt funcții derivabile și strict crescătoare, P este strict concav și Q strict convex. Prin urmare, se poate presupune $A\leqslant 1$. De notat că s-a presupus că reprezentantul este refractar la risc și întreprinzătorul este neutru față de risc.

Fără a se pierde generalitatea se va face convenția că

$$P(0)=Q(0)=0$$

Întreprinzătorul primește ce a rămas din venit după compensarea reprezentantului. Se presupune că cîștigul său este egal cu ceea ce primește, adică:

$$U_0=C-R(C)$$

În acest joc, întreprinzătorul alege o funcție de compensație R — prima fază — apoi reprezentantul alege o acțiune după ce află cum este R — a doua fază. Cîștigul sperat ce rezultă pentru întreprinzător este:

$$V_0=A(1-w_1)-(1-A)w_0 \quad (6.1)$$

și pentru reprezentant

$$V_1=AP(w_1)+(1-A)P(w_0)-Q(A) \quad (6.2)$$

Strategia întreprinzătorului este funcție de compensația R, iar strategia reprezentantului este o transformare α de la funcțiile de compensare la activități:

$$A=\alpha(R)$$

Un echilibru al jocului este o pereche de strategii (R^*, α^*), astfel ca:

1. R^* maximizează V_0 dat de α^*
2. $\alpha^*(R^*)$ maximizează V_1 dat de R^*

O condiţie suplimentară necesară la echilibru este că pentru fiecare R (nu numai pentru R*), $\alpha^*(R)$ maximizează cîştigul sperat R dat. (Astfel, se cere ca echilibrul să fie perfect; în acest joc în două faze un astfel de echilibru este numit echilibrul Stackelberg.) Se va scrie $A^*=\alpha^*(R^*)$.

Este realist să se impună două restricţii asupra compensaţiilor. Prima este aceea datorită căreia întreprinzătorul nu poate impune în mod arbitrar penalizări mari reprezentantului; cu alte cuvinte, compensaţiile sînt astfel stabilite încît pierderea reprezentantului este limitată în jos. A doua restricţie exprimă condiţia că reprezentantul este liber să refuze să facă parte din relaţie (adică să participe la joc). Pentru aceasta, w_0 şi w_1 trebuie să fie astfel stabilite încît să-i permită reprezentantului să obţină un cîştig sperat minim. În scopurile propuse în acest articol, este suficient să se impună o restricţie de primul tip; adăugarea celei de a doua restricţii ar complica expunerea, dar nu ar modifica rezultatele în mod esenţial.

Pentru a exprima prima restricţie, se poate presupune că compensaţiile sînt limitate inferior (şi astfel funcţia P este finită în tot domeniul); se poate presupune deci, că ele nu sînt negative.

$$(w_0, w_1)\geqslant 0$$

Spaţiul limitat nu permite o analiză completă a acestui joc, în cadrul articolului de faţă. Se poate uşor verifica din relaţia (6.2) că dacă $w_0=w_1$, reprezentantul nu va fi stimulat să lucreze, adică $\alpha^*(w, w)=0$. Tot din (6.2) se poate vedea că dacă:

$$Q'(0)\geqslant P(1)$$

atunci $\alpha^*(w_0, w_1)=0$ pentru toate valorile w_0 şi w_1 cuprinse între 0 şi 1; în acest caz un singur echilibru are $R^*=(0, 0)$ şi $\alpha^*=0$. Dacă însă:

$$Q'(0)\geqslant P(1) \qquad (6.3)$$

atunci echilibrul este caracterizat de:

$$0=w_0^*<w_1^*<1 \qquad (6.4)$$

de asemenea, $\alpha^*(0, w_1)$ este strict crescătoare în w_1 oricînd $\alpha^*(0, w_1)$ este strict în domeniul 0 . . . 1. Acest caz va fi discutat în continuare.

O pereche $(\hat{R}, \hat{A})$ este *eficientă* (eficienţă Pareto optimă) dacă nici o altă pereche (R, A) nu produce pentru fiecare jucător acelaşi cîştig sperat şi, cel puţin pentru un jucător, mai mult.

Este uşor de observat că pentru acelaşi nivel de efort reprezentantul preferă funcţia de compensaţie $(\bar{w}, \bar{w})$ faţă de funcţia (w_0, w_1), unde

$$\bar{w}=Aw_1+(1-A)w_0$$

în timp ce întreprinzătorul este indiferent în adoptarea uneia sau a alteia din cele două funcţii (se reaminteşte că reprezentantul este refractar riscului şi întreprinzătorul este neutru faţă de risc).

Deci, dacă $[(\hat{w}_0, \underset{\wedge}{w}_1), \hat{A}]$ este eficient, $\check{w}_0$=atunci $\hat{w}_1$.

Împreună cu (6.4) aceasta demonstrează că un echilibru nu este suficient. Există, desigur, multe perechi $[(\underset{\wedge}{w}, \hat{w}), \hat{A}]$ eficiente; pentru $0<\hat{A}<1$, ele sînt caracterizate de condiția:

$$P'(\hat{w})=Q'(\hat{A})$$

O excepție apare dacă reprezentantul este neutru față de risc și suficient de bogat. În acest caz, un echilibru eficient se obține dacă întreprinzătorul vinde reprezentantului un „drept" asupra întreprinderii, adică reprezentantul plătește întreprinzătorului o anumită sumă, și atunci păstrează întregul rezultat. (Este ușor de văzut că aceasta este echivalent cu a face negativă compensația pentru faliment și compensația pentru succes cu o unitate mai mare decît compensația pentru faliment).

7. Riscul moral și concurența: modelul participației

În organizațiile complexe rezultatele activității organizaționale depind de obicei de acțiunile mai multor factori de decizie ca și de aspectele stohastice ale restricțiilor organizației și este adesea dificil dacă nu imposibil să se identifice în mod separat contribuțiile individuale ale factorilor de decizie și ale factorilor exteriori (restricții) în obținerea rezultatelor. Cu descentralizarea factorilor de decizie sînt asociate imperfecțiuni serioase în dirijarea informațiilor și activităților individuale.

Rezultatului activităților diferiților factori de decizie, care ar putea fi denumite fenomene de concurență se adaugă fenomenul riscului moral care este deja prezent în relația întreprinzător-reprezentant. În această secțiune se va ilustra combinația dintre fenomenele de concurență și risc moral printr-un exemplu simplu care a fost denumit mai înainte „modelul participației".

Modelul general al participației este un joc în care jucătorii — numiți „parteneri" — împart un rezultat care depinde strîns de activitățile partenerilor și de restricțiile stohastice. Nici un partener nu poate dirija complet informațiile, activitățile sau restricțiile celorlalți.

Structura informației este la fel ca în teoria echipelor, dar modelul ține în mod explicit de scopurile conflictuale parțiale ale factorilor de decizie. Se presupune că cîștigul fiecărui partener depinde direct de propria sa activitate și de mărimea aportului său la rezultat (stohastic).

Se va relua modelul din Secțiunea 2, cu multe modificări.

Mai întîi, se consideră că nu mai există un „centru", astfel că singurii factori de decizie sînt directorii filialelor N. Se consideră, de asemenea, că nu mai există resurse care să fie alocate de la centru. În al doilea rînd se va considera că în funcțiile generale care exprimă rezultatul, venitul total al firmei depinde strîns de activitățile directorilor filialelor și de restricția X; nu există venituri separate ale filialei. Astfel, ecuația (2.2) ia forma:

$$C=G(A_1 \ldots A_N, X) \quad (7.1)$$

Fiecare director de filială i este compensat conform unei funcţii R_i, al cărei argument este venitul realizat al firmei; astfel, compensaţia este:

$$W_i = R_i(C) \quad (7.2)$$

Compensaţia totală nu poate depăşi venitul:

$$\Sigma_i R_i(C) \leqslant C \quad (7.3)$$

Pentru moment, se poate considera că nu este exclus ca compensaţiile să fie uneori negative. De asemenea, nu se cere ca totalul compensaţiei să fie egal cu venitul, cu toate că acesta va fi cazul obişnuit. Deoarece factorii de decizie îşi împart venitul, care depinde de eforturile lor comune, această organizaţie se va numi o *asociaţie*, iar factorii de decizie *parteneri*.

Se presupune că cîştigul directorului este:

$$U_i = P_i(w_i) - Q_i(A_i) \quad (7.4)$$

unde P_i este cîştigul său în funcţie de compensaţie iar Q_i este „pierderea" sa în funcţie de activitate.

Strategia partenerului i este funcţia sa de decizie D_i, care determină activitatea sa şi ca o funcţie de informaţia Y:

$$A_i = D_i(Y_i) \quad (7.5)$$

Distribuţia probabilităţii conexate variabilelor aleatorii X, $Y_1 \ldots Y_N$, este dată. Această distribuţie a probabilităţii şi ecuaţiile (7.1) ... (7.5) determină *jocul asocierii*, în care cîştigul sperat:

$$V_i(D_1 \ldots D_N) = EU_i. \quad (7.6)$$

pentru fiecare partener i depinde de strategiile $D_1 \ldots D_N$ ale partenerilor. Se presupune că partenerii aleg strategiile lor simultan. *Echilibrul* este o combinaţie de strategii astfel ca nici unui partener să nu-i poată creşte cîştigul sperat prin alegerea unilaterală a propriei sale strategii. De notat că funcţia de compensaţie face parte din regulile jocului.

Pentru a ilustra ineficienţa tipică a echilibrelor în jocul asociaţiei se va considera un exemplu în care nu există semnale de informaţie (variabilele aleatorii $Y_1 \ldots Y_N$ sînt constante), activitatea fiecărui partener i este un număr real pozitiv (denumit „efort"), iar partenerii împart venitul egal:

$$W_i = \frac{1}{N} G(A_1 \ldots A_N, X) \quad (7.7)$$

Se presupune că, pentru fiecare X, G este o funcţie de două ori derivabilă strict concavă şi crescătoare a activităţilor A_i.

Se presupune, de asemenea, că pentru fiecare i, P_i este o funcţie de A_i, de două ori derivabilă strict concavă şi crescătoare şi Q_i este o funcţie de A_i, de două ori derivabilă, strict convexă şi crescătoare. Se notează cu A activităţile $(A_1 \ldots A_N)$, cu G_i' derivata parţială a lui G_i

în raport cu A_i și cu P'_1 și Q'_1 derivatele lui P_i și respectiv Q_i. Condiția primordială pentru un echilibru este:

$$\frac{\partial V_i}{\partial A_i} = =E\left[\left(\frac{1}{N}\right)P'_i(C)\,G'_i(A, X)\right]-Q'_i(A_i)=0 \qquad (7.8)$$

Fie A^* un echilibru. Se consideră o mică variație a lui A, sau $dA= =(dA_1, \ldots dA_N)$, de la echilibrul A^*. Variația corespunzătoare a lui V_i este:

$$dV_i=E\left[\left(\frac{1}{N}\right)P'_i(C)\sum_{j=1}^{N} G'_j(A^*, X)\,dA_j\right]-Q'_i\,(A^*_i)dA_j$$

$$=E\left[\left(\frac{1}{N}\right)P'_i(C)\sum_{j\neq 1} G'_j(A^*, X)\,dA_j+\right. \qquad (7.9)$$

$$+\left\{E\left[\left(\frac{1}{N}\right)P'(C).G'_i(A^*, X)\right]-Q'_i(A^*_i)\right\}\,dA_i$$

Din condiția de la (7.8) termenul dintre acolade din (7.9) este 0. Primul termen din membrul drept al egalității (7.9) este pozitiv și dacă fiecare dA_j este pozitiv, dV_i va fi și el pozitiv. Astfel, plecînd de la un echilibru A^*, o creștere *simultană* (suficient de mică) a tuturor eforturilor duce la o creștere a cîștigului sperat al fiecărui partener.

Astfel, un echilibru nu este eficient. Primul termen din membrul drept al expresiei (7.9) reprezintă „extrema pozitivă“ care la o creștere a efortului partenerului i, determină un produs pentru alți parteneri, dar pe care partenerul i nu o ia în considerare în calculul propriului său cîștig.

Exemplul particular folosit schematic în text are în vedere doi parteneri. Venitul C este egal cu 1 sau 0 și:

Prob $(C=1)=\min(A_1+A_2, 1)$

$P_i\left(\frac{1}{2}\right)=1,\ P_i(0)=0$

$Q_i(A_i)=qA_i^2$

(Pentru calculul numeric, $q=2{,}8$). Este ușor de verificat că dacă $q\geqslant 1$ există un singur echilibru în care

$$A_1=A_2=\frac{1}{2q}$$

$$V_i(A_1, A_2)=\frac{3}{4q}\equiv u^*$$

Dacă $q>2$ echilibrul nu este eficient. De exemplu, dacă $q>2$, atunci o pereche eficientă de efort care domină echilibrul este:

$$A_1=A_2=\frac{1}{q}$$

$$V_i(A_1, A_2)=\frac{1}{q}\equiv\hat{u}$$

Pentru alte detalii a se vedea (Radner, Myerson și Maskin 1983).

8. Relaţii pe termen lung

În cele mai economice organizaţii membrii au relaţii cu interacţiuni repetate. Chiar cu creşterea mobilităţii actuale a vieţii guvernamentale şi de corporaţie, aceste relaţii pot continua pentru ani sau chiar zeci de ani. Aceasta creează membrilor ocazii repetate să evalueze calităţile membrilor asociaţi şi să stabilească reputaţiile lor, chiar dacă nu-şi pot observa reciproc activităţile direct şi precis. Există un dicton despre comportarea organizaţională conform căruia *relaţiile pe termen lung pot fi utilizate pentru reducerea ineficienţei care rezultă din riscul moral în relaţiile pe termen scurt.*

În această secţiune se va examina cum apropierea jocului teoretic de teoria organizării poate contribui la înţelegerea acestui fenomen.

A modela situaţia într-o manieră exactă presupune că unul din jocurile descrise în secţiunile anterioare este repetat de un număr finit sau infinit de ori. Evoluţia restricţiei stohastice va fi descrisă ca un proces stohastic. În mod suplimentar, pot exista unele variabile de stare care se schimbă *din interior* la fiecare repetare a jocului.

Se presupune în continuare că fiecare jucător obţine unele cîştiguri la fiecare repetare a jocului (ca rezultat al activităţii şi rezultatului în timpul jocului) şi că fiecare cîştig (sperat) al jucătorului pentru fiecare secvenţă a repetării jocului este suma cîştigurilor (sperate) din timpul repetării şi cîştigurile posibile viitoare pierdute.

În final, pentru a completa descrierea formală a structurii jocului trebuie să se specifice setul de strategii utilizabil de fiecare jucător. O *strategie secvenţială* a jucătorului este o succesiune de reguli care determină activitatea (activităţile) sale în timpul fiecărei repetări ca o funcţie a *istoriei* informaţiilor sale pînă în momentul cînd activitatea sa este considerată.

Construcţia care rezultă este numită un *joc secvenţial*. Dacă nu există nici o variabilă de stare care se modifică de la o repetare la alta şi dacă restricţiile succesive sînt independente şi identic distribuite, atunci jocul secvenţial este denumit *joc repetat* sau *superjoc*.

Din descrierea setului de strategii secvenţiale utilizabil de un jucător, este clar că el creşte rapid în dimensiune şi complexitate cu numărul repetărilor din jocul secvenţial. În scopul de a obţine o idealizare matematică convenabilă a relaţiilor pe termen foarte lung, se va presupune (dacă nu se precizează altfel) că numărul repetărilor este infinit. (Deoarece „infinit" este o bună aproximare pentru „foarte lung" în acest caz, este necesară o cercetare atentă.)

Deoarece setul de strategii secvenţiale utilizabil de fiecare jucător este foarte mare, nu trebuie să surprindă că:

— Într-un joc secvenţial cu infinit de multe repetări există în mod obişnuit infinit de multe echilibre.

Această afirmaţie are importante implicaţii pentru înţelegerea relaţiilor pe termen lung ca şi pentru limitările teoriei jocului în cazul dezvoltării lui.

9. Concluzii

În această secțiune se va prezenta un rezumat al analizelor anterioare.

În rezumatul care urmează sînt prezentate schematic principalele concluzii ale acestor analize.

1. Echilibrele organizațiilor descentralizate sînt de obicei ineficiente, comparativ cu ceea ce ar trebui teoretic realizat cu o distribuție dată a informației printre diferiții factori de decizie.

2. Ineficiența echilibrelor este cauzată de variate probleme de stimulare care rezultă din conflictele de interese dintre factorii de decizie și/sau dintre factorii de decizie și organizator. Aceste probleme de stimulare pot fi clasificate în trei categorii:

a) denaturarea
b) riscul moral
c) concurența

3. Denaturarea poate fi remediată prin mecanisme de compensație adecvate, „balanța de bugete“ furnizată nu este necesară (mecanismele Groves-Clarke-Vickrey).

4. Ineficiența datorată riscului moral poate fi remediată în echilibrele pe termen lung, în măsura în care jucătorii nu privesc prea aproape, adică în măsura în care ei nu reduc foarte mult cîștigul viitor (jocurile întreprinzător-reprezentant).

5. Concurența și riscul moral, împreună, pot fi numai parțial remediate în echilibrele pe termen lung cu jucători cu vederi largi, dacă nu există calități individuale observabile în public. (Jocurile de asociere.)

6. Eficiența poate fi îmbunătățită în echilibrele pe termen lung cu jucători cu vederi largi dacă ei nu sînt prea „lacomi“ (Echilibrele aproximative).

7. Jocurile pe termen lung au, în mod obișnuit, infinit mai multe echilibre.

10. Note bibliografice

Iacob Marschak a prezentat un program de cercetări asupra teoriei organizării în lucrarea sa „Cu privire la teoria economică a organizării și informației“ (J. Marschak 1954) și a definit cazul special al unei *echipe* (J. Marschak 1955). Teoria echipelor a fost dezvoltată în continuare (Radner 1961) și (Radner 1962) utilizîndu-se o tratare probabilistică a incertitudinii și informației care a fost preluată din teoria deciziei statistice. Altă contribuție la teoria descentralizării, în aceeași idee, a fost adusă de T. A. Marschak (1959). O elaborare sistematică a teoriei jocurilor este dată de Marschak și Radner (1972) împreună cu un material introductiv asupra teoriei unipersonale a deciziei și informației (Pentru un rezumat mai scurt a se vedea Radner, 1972 a). Materialul din secțiunea 2 și 3 este

bazat pe Radner (1972 b) și Groves și Radner (1972). Aceste ultime două lucrări pun, de asemenea, la dispoziție rezultate din teoria asimptotică a alocării resurselor în „echipe mari", generalizări și extinderi ale acestor rezultate care au fost apoi elaborate de Arrow și Radner (1979) și de Groves și Hart (1982). Independent, Witsenhausen (1968) a formulat și analizat o problemă teoretică de control dinamic cu o structură de echipă a informației, continuată cu contribuțiile lui Ho și Chu (1974) și a altora.

Într-o măsură destul de diferită, Huwicz (1960) a iniția o analiză sistematică a descentralizării informaționale utilizînd o descriere neprobabilistică a incertitudinii și informației iar în 1972 el a arătat ce probleme de stimulare ar putea conduce la ineficiență în mecanismele informaționale de alocare descentralizată. Pe de altă parte, în lucrarea sa de doctorat, Groves (1969, 1973) a introdus considerarea stimulentelor în modelele cu o structură de echipă a informației și a arătat că în cazurile particulare informația reală și decizia eficientă ar putea fi hotărîte de mecanisme de compensare corespunzătoare. Independent, Clarke (1971) a propus un mecanism similar de compensație; de fapt, acest tip de mecanism a fost, de asemenea, propus mai înainte de Vickrey (1961) în contextul modelelor licitației. O examinare a mecanismelor Groves-Clarke-Vickrey poate fi găsită în (Green și Laffont, 1979) și (Groves și Ledyard, 1983)

Modelul pe care Groves l-a analizat în dizertația sa a inclus modelul întreprinzător-reprezentant ciclic ca un caz special, dar studii independente în această problemă au fost făcute de Arrow (1963, 1965), Ross (1973) și Mirrless (1974). O tratare mai puțin formală a fost sugerată de Alchian și Demsatz (1972) în discuția lor asupra teoriei firmei. Pentru lucrări ulterioare a se vedea (Hurwicz și Shapira, 1978), (Mayers și Smith, 1981) și (Grossman și Hart, 1983), precum și referințele citate în lucrările lor (spațiul nu permite enumerarea lor în cadrul lucrării de față).

Un premergător al literaturii despre modelul întreprinzător-reprezentant a fost Simon (1951).

Proprietățile echilibrului superjocului apar în cazul jocurilor repetate în condiții în care 1) jucătorii pot dirija activitățile altor jucători după fiecare repetare a jocului („monitorizare perfectă") și 2) jucătorii nu își reduc cîștigurile lor viitoare („fără reducere").

În acest caz, direcția vectorilor cîștigului sperat al echilibrului superjocului este aceeași cu direcția posibilă a vectorilor plăți individuale în jocul repetat. (Aceasta este așa-numita teoremă Falk.) Aceeași concluzie poate fi trasă pentru echilibrul *perfect* al superjocului; acest rezultat este datorat lui Anmann și Shapley (nepublicat) și lui A. Rubinstein; pentru referințe și rezultate în legătură cu această problemă vezi (Rubinstein, 1979 b). Cazul dirijării perfecte cu reducerea cîștigului nu a fost explorat suficient.

Din păcate, condiția unei perfecte conduceri este exclusă (fără costuri suplimentare) de structura informațională a relațiilor repetate întreprinzător-reprezentant. Pentru cazul nereducerii cîștigului Radner (1981 a) a arătat că pentru superjocurile întreprinzător-reprezentant suficient de lungi dar finite se poate susține eficiența aproximată, prin

echilibrul aproximat. Jocurile întreprinzător-reprezentant, particular infinite, au fost analizate într-un spirit similar, tot pentru cazul nereducerii cîștigului, de Rubinstein (1979 a) și de Rubinstein și Yaari (1983).

Rezultatul în jocurile infinite întreprinzător-reprezentant cu reducerea cîștigurilor prezentate în secțiunea 8.1 se bazează pe (Radner 1981 b). O aplicație a acestor idei (în cazul reducerii) la problema reglării cîștigului public se găsește în Linhart (1983). În această aplicație există stări variabile de la o perioadă la alta astfel că jocul nu este strict repetat. Această circumstanță, care are șansa să caracterizeze cele mai multe aplicații, reclamă o analiză mai adîncă decît în cazul jocului repetat.

Materialul din secțiunea 8.2 despre jocurile de asociere repetate, este bazat pe lucrarea lui Radner (1981 c) pentru cazul nereducerii cîștigului și pe lucrarea lui Radner, Myerson și Maskin (1983) pentru cazul reducerii lor. Unele discuții asupra jocurilor repetate de asociere pot fi găsite în lucrarea lui Holmstrom (1982).

BIBLIOGRAFIE

1. Alchian, A. A., și H. Demsetz: 1972, Production Information Costs, and Economic Organization, *Amer. Econ. Rev.* 62, 777—795.
2. Arrow, K. J.: 1963, Uncertainty and the Welfare Economics of Medical Care, *Amer. Econ. Rev.* 53, 941—973.
3. Arrow, K. J.: 1965, *Aspects of the Theory of Risk-Bearing,* Yrjo Johanssonin Saatio, Amsterdam, Lecture 3.
4. Arrow, K. J. și R. Radner: 1979, Allocation of Resources in Large Teams, *Econometrica* 47, 361—385.
5. Clarke, E.: 1971, Multipart Pricing of Public Goods, *Public Choice* 11, 17—33.
6. Green, J., și J. J. Laffont: 1979, *Incentives in Public Decision-Making.* North-Holland Publishing Company, Amsterdam.
7. Grossman, S. J., și O. D. Hart: 1983, An Analysis of the Principal-Agent Problem, *Econometrica* 51, 7—45.
8. Groves, T. F.: 1969, The Allocation of Resources under Uncertainty: The Informational and Incentive Roles of Prices and Demands in a Team, *NSF Technical Report No1,* Center for Research in Management Science, University of California, Berkeley (nepublicat).
9. Groves, T. F.: 1973, Incentives in Teams, *Econometrica* 41, 617—631.
10. Groves, T. F., și S. Hart 1982, Efficiency of Resource Allocation by Uniformed Demand, *Econometrica* 50, 1453—1482.
11. Groves, T. F., și J. Ledyard: 1977, Optimal Allocation of Public Goods: A Solution to the „Free Rider“ Problem, *Econometrica* 45, 783—309.
12. Groves, T. F., și J. Ledyard: 1983, The Allocation of Public Goods: An Overview, in T. F. Groves et al (eds.), *Allocation of Resources: Essays in Honor of Leonid Hurwicz,* (în curs de apariție).
13. Groves, T. F., și M. Loeb: 1975, Incentives and Public Inputs, *J. of Public Economics* 4, 211—226.
14. Groves T. F., și R. Radner: 1972, Allocation of Resources in a Team, *J. of Economic Theory* 3, 415—44.
15. Ho. Y. C., și K. C. Chu: 1974, Information Structure in Dynamic Multi-Person Control Problems, *Automatica* 10, 341—351.
16. Holmstrom, B.: 1982, Moral Hazard in Teams, *Bell J. of Econ.* 13, 324—340.
17. Hurwicz, L.: 1960, Optimality and Informational Efficiency in Resorce Allocation Processes, in K. J. Arrow, et al (eds.), *Mathematical Methods in the Social Sciences,* Stanford University Press. Stanford. Calif., 1960 m pp. 27—46.
18. Hurwicz, L.: 1972, On Informationally Decentralized Systems, in C. B. McGuire and R. Radner (eds.), *Decision and Organization,* North-Holland Publishing Co., Amsterdam, 1972, pp. 297—336.

19. Huricz, L., şi L. Shapiro: 1978 Incentive Structures Maximizing Residual Gain under Incomplete Information, *Bell J. of Econ.* 9, 180—191.
20. Linhart, P.B.R. Radner, şi F. W. Sinden: 1983, A Sequential Principal-Agent Approach to Regulation, Bell Laboratories Discussion Paer, 1983 (nepublicat).
21. Marschak, J.: 1954, Towards an Economic Theory of Information and Organization, in R. M. Thrall, et al (eds.), *Decision Processes, Wiley,* New York. 1954, pp.
22. Marschak, J.: 1955, Elements for a Theory of Teams, *Management Science* 1, 127—137.
23. Marschak, J. and R. Radner: 1972, *Economic Theory of Teams,* Yale University Press, New Haven.
24 Marschak, T. A.: 1959, Centralization and Decentralization in Economic Organizations, *Econometrica* 27, 399—430.
25. Mayers, A. B., şi C. W. Smith: 1981, Contractual Provisions, Organizational Structure, and Conflict Control in Insurance Markets, *J. of Business* 407—434.
26. Mirrlees, J.: 1974, Notes on Welfare Economics, Information, and Uncertainty, in Balch et al (eds.) *Essays on Economic Behavior under Uncertainty,* North-Holland Publishing Co., Amsterdam, 1974.
27. Myerson, R. B.: 1983, Bayesian Equilibrium and Incentive Compatibility: An Introduction, Discussion Paper No. 548, Center for Mathematical Studies in Economics and Management Science, Northwestern University Evanston, Illinois.
28. Radner, R.: 1961, The Evaluation of Information in Organizations, in *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability,* University of California Press. Berkeley, 1961, Vol. 1, pp. 491—530.
29. Radner, R.: 1962, Team Decision Problems, *Annals of Mathematical Statistics* 33, 857—881.
30. Radner, R.: 1972a, Teams, in C. B. McGuire şi R. Radner (eds.), *Decision and Organization,* North-Holland Publishing Co., Amsterdam, 1972, pp. 189—215.
31. Radner, R.: 1972b, Allocation of a Scarce Resource under Uncertainty: An Example of a Team, in C. B. McGuire and R. Radner (eds.), *Decision and Organization,* North-Holland Publishing Co., Amsterdam, 1972, pp. 217—236.
32. Radner, R.: 1981a, Monitoring Cooperative Agreements in a Repeated Principal-Agent Relationship, *Econometrica* 49, 1127—1148.
33. Radner, R.: 1981b, A Repeated Principal-Agent Game with Discounting, Bell Laboratories Discussion Paper, Murray Hill, N. J., May 1981 (va apare în *Econometrica*).
34. Radner, R.: 1981c, Optimal Equilibria in Some Repeated Partnership Games with Imperfect Information, Bell Laboratories Discussion Paper, Murray Hill, N. J., May 1981 (nepublicat).
35. Radner, R., R. Myerson, şi E. Makin: 1983, An Example of a Repeated Partership Game with Discounting and with Uniformly Inefficient Equilibria, Bell Labortories Discussion Paper, Murray Hill, N. J., June 1983 (nepublicat).
36. Ross, S.: 1973, The Economic Theory of Agency: The Principal's Problem, *Amer. Econ. Rev.* 63, 134—139.
37. Rubinstein, A.: 1979a, Offenses that May Have Been Committed by Accident — an Optimal Policy of Retribution, in S. J. Brams et al (eds.), *Applied Game Theory,* Physica-Verlag, Wurzburg, 1979.
38. Rubinstein, A.: 1979b, Equilibrium in Supergames with the Overtaking Criterion, *J. of Econ. Theory,* 21, 1—9.
39. Rubinstein, A., şi M. Yaari: 1983, Repeated Insurance Contracts and Moral Hazard, *J. of Econ. Theory.*
40. Simon, H. A.: 1953, A Formal Theory of the Employment Relationship, *Econometrica* 19, 293—305.
41. Vickrey, W.: 1961, Counterspeculation, Auctions, and Competitive Seald Tenders, *Journal of Finance* 16, 8—37.
42. Witsenhausen, H. S.: 1968, A Counterexample in Stochastic Optimum Control, *SIAM J. Control* 6, 131—147.

P.6. REGLAREA AUTOMATĂ ÎN ŞI PENTRU BIOSISTEME

V. A. VIKTOROV
Institutul de instrumente medicale, Moscova, U.R.S.S.

V. N. NOVOSELŢEV
Institutul de ştiinţe ale conducerii, Moscova, U.R.S.S.

V. I. ŞUMAKOV
Institutul de transplantologie şi organe interne artificiale, Moscova, U.R.S.S.

Rezumat: Se consideră o gamă largă de probleme în reglarea biosistemelor. Se studiază modelele şi mecanismele de reglare în organism şi se face o analogie între organism şi maşină. Sînt clasificate şi trecute în revistă mijloacele tehnice folosite ca instrumente auxiliare pentru funcţiile de bază ale organismului. Se discută asupra problemelor biotehnice proprii proiectării complexelor pentru diagnosticare, terapeutică şi substituirea pînă a celor din urmă funcţii ale corpului. Se formulează o problemă de fiziologie inginerească necesară analizei funcţionării comune a sistemelor fiziologice şi mijloacelor tehnice.

Se consideră problema reglării cordului artificial, ca o problemă tipică de reglare automată pentru organism.

Cuvinte cheie: Cibernetica biologică; biologie; biomedical; sisteme om-maşină; sisteme medicale; modelare; modele fiziologice.

Introducere

Reglarea în şi pentru sisteme biologice a devenit în ultimii zece ani o arie majoră de aplicaţie pentru metodele şi mijloacele teoriei reglării automate. Un cerc larg de specialişti cunoaşte foarte bine nivelul la care a ajuns reglarea funcţiilor organismului, în biomedicină, incluzînd cîteva inovaţii recente în instrumentaţia biomedicală pentru organele interne artificiale. Mai puţin cunoscute sînt realizările în domeniul reglării automate la noile medii de locuit ale omului în spaţiu, în regiuni arctice şi subacvatice. Un orizont larg este deschis biotehnologiei producerii de alimente folosind populaţii de microorganisme supravegheate, pentru fabricarea industrială a produselor proteice. În general teoria reglării automate este pregătită pentru o asemenea expansiune a domeniilor sale de aplicaţie. Pînă în prezent nu s-a făcut simţită necesitatea dezvoltării unor tehnici principial noi. Aceasta se explică probabil prin faptul că interacţiunea dintre echipamentul de reglare şi biosisteme este realizat de cele mai multe ori la cel mai scăzut nivel de organizare al biochimiei, fiziologiei şi ecologiei, în timp ce rolul major este jucat de procesele de schimb de substanţă şi energie dintre organisme şi mediu. Extinderea formalizării în descrierea biosistemelor, obţinută la aceste nivele, prin unirea eforturilor teoriei reglării şi ştiinţelor biologice este în general suficientă pentru folosinţa efectivă a mijloacelor tehnice. Pentru a realiza aceasta, au fost necesare doar unele modificări ale tehnicilor şi aparaturii disponibile. Mai mult, este foarte posibil ca progresele obţinute în domeniul reglării biosistemelor să poată să ne furnizeze o înţelegere mai profundă a fenomenelor vitale.

Acestea sînt, în primul rînd, procesele de reglare a substanţelor şi de schimb de energie în biosisteme.

Totuşi, nu este deloc sigur că această situaţie favorabilă a metodelor teoriei reglării automate va face faţă şi în viitor cînd automatica se va ocupa cu probleme de reglare a proceselor, de schimb controlat şi de prelucrare a datelor, la nivele mai ridicate de organizare a biosistemelor, ca de exemplu, prelucrarea informaţiei în sistemul nervos central al unui organism. În prezent, nu există indicaţii că tehnicile disponibile pot ajuta la obţinerea unui progres substanţial în acest domeniu.

S-a constatat deja că problemele de reglare, chiar şi la nivelele joase de organizare a biosistemelor, au dus la anumite modificări a mijloacelor cu care lucrează teoria reglării automate, ceea ce s-a datorat naturii specifice a biosistemelor. Această specialitate necesită adesea separarea problemelor de reglare în biosisteme, de cele de aparatură de reglare autonată, apropiindu-le pe primele de conducerea structurilor economice şi sociale.

Din punct de vedere al teoriei reglării automate, trăsătura specifică, de bază a biosistemelor, care le deosebeşte de alte obiecte comandate, este mecanismul de reglare specific acestora, şi scopurilor pe care le urmăresc prin reglare. Acest mecanism răspunde activ faţă de fiecare încercare de reglare din exterior. De aceea rezultatul reglării este întotdeauna determinat de interacţiunea dintre reglarea naturală şi artificială.

Vom arăta acestea printr-un exemplu simplu. Producţia de căldură naturală într-un organism este insuficientă pentru a opera în condiţii de frig excesiv şi vom folosi de aceea un sistem de căldură artificial. Acest sistem cuprinde un set de tuburi care acoperă corpul şi prin care circulă apă caldă (Koşceev, 1981). Ne putem da seama că, cu cît vine mai multă căldură de la încălzitor, cu atît temperatura corpului este mai mare. Nu se întîmplă însă întotdeauna aşa. Creşterea temperaturii apei peste

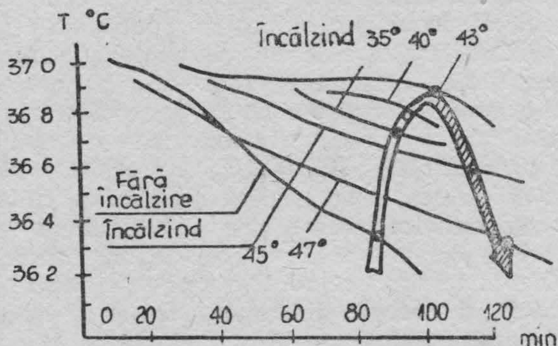


Figura 1. Date experimentale referitoare la căldura furnizată artificial organismului: Temperatura corpului funcţie de timp este arătată (testul a fost efectuat pe 8 persoane) la temperatura mediului de 25°C. Cel mai bun rezultat a fost obţinut cu temperatura de încălzire de 43°C. Creşteri mai mari de temperatură determină înrăutăţirea stării termale a corpului.

un anumit punct, duce mai repede la o descreştere decît la o creştere a temperaturii corpului (Fig. 1). La acest punct (t=43°C) mecanismele proprii de reglare a temperaturii corpului „decid“ că organismul este supra-

încălzit și se activează o intensă pierdere de energie. Acest mecanism poate lucra mai repede decît încălzirea și corpul se răcește în loc să se încălzească.

Problema analizei, înțelegerii și descrierii mecanismelor naturale de reglare în biosisteme, este de asemenea importantă, ca o necesitate anterioară pentru soluționarea problemelor tehnice de reglare a biosistemelor. Problema unei mai bune reglări a biosistemului se poate trata la fel ca cea a organizării unei interacțiuni adecvate între mecanismele de reglare naturale și artificiale. În acest sens analizele și descrierile mecanismelor de reglare naturală a biosistemelor sînt strîns legate cu un alt domeniu, al teoriei moderne a reglării automate, acela al modelării și simulării.

1. Reglarea în biosisteme

1.1. *Structura organismului*

Un organism viu cuprinde o mare varietate de elemente, fiecare destinat pentru realizarea unei anumite funcțiuni.

Sistemele de receptori ale organismului sesizează stadiul mediului înconjurător și al celui intern. Folosind informația furnizată de receptori, mecanismele centrale ale corpului elaborează un program de comportare pentru organism care este îndeplinit de sistemele sale de acționare.

Într-un organism activitatea de acționare este realizată întotdeauna prin mișcare. Astfel, mișcarea arcușului dă naștere muzicii, în timp ce mișcarea mușchilor respiratorii dă oxigen corpului.

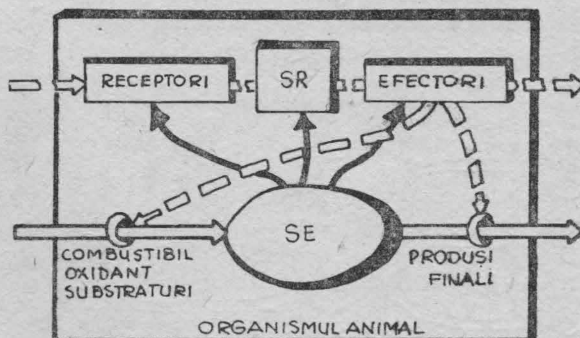


Figura 2. Schema bloc a proceselor majore într-un organism viu: SR — partea cibernetică (sistemul de reglare); SE — partea metabolică (sistemul energetic).

O mișcare de orice tip este asociată cu consum de energie. Energia este sintetizată și stocată în organism datorită consumului exterior de combustibil — lipide și hidrocarburi. Pentru arderea combustibilului este necesar un oxidant. Oxidantul — oxigenul — provine tot din exterior. Oxidarea combustibilului dă energie pentru funcțiile vitale ale corpului,

pentru activitatea receptorilor, a sistemului nervos central și a sistemelor de acționare.

Produșii finali ai metabolismului, nefolositori pentru organism, sînt eliminați în exterior. Fig. 2 arată interacțiunea dintre elementele organismului.

Partea de sus a figurii conține lanțul: „receptori — sistem nervos central — efectori". Aceasta este partea cibernetică a organismului. Ea este răspunzătoare pentru capacitatea senzorială, de stocare, de producere și de utilizare a informației pentru a asigura activitatea organismului.

Scopul general al acestei conexiuni este să asigure integritatea organismului, să mențină structura și funcționarea energetică corectă (echilibrul metabolic), ceea ce este arătat în jumătatea de jos a figurii.

1.2. *Sistemul energetic al organismului*

Este bine cunoscut faptul că un organism este un sistem deschis, efectuînd schimb de substanțe și energie cu mediul său ambiant.

Aceleași procese se desfășoară în sistemele energetice a tuturor sistemelor vii, de la cele mai inferioare organisme pînă la cele mai dezvoltate (superioare). Astfel, o celulă vie execută următoarele funcții (Guyton, 1971):

— obținerea substanțelor din mediu (transportul lor prin membranele celulelor și digestia în lizozomi);

— extragerea energiei din aceste substanțe (în mitocondrii);

— sintetizarea substanțelor necesare și formarea structurii celulare (în reticulul endoplasmic și complexul Golgi);

— excreția produșilor finali (prin membrane).

În organismele superioare sînt efectuate aceleași procese în sistemul energetic printr-un set mai complicat de structuri și organe speciale.

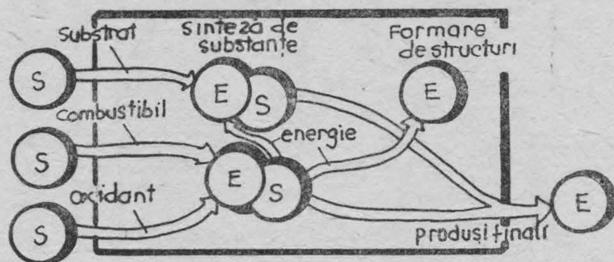


Figura 3. Procese majore în sistemul energetic al organismului. S sînt sursele iar E sînt orificiile de evacuare (scurgerile).

Funcțiile majore ale unui organism, ca un sistem deschis sînt trecute în tabelul 1.

Substanțele intrate în organism sînt: substraturi, combustibili și oxidanți. Figura 3 arată destinația acestor substanțe în cadrul sistemului energetic. Scopul final pentru toate procesele din acest sistem este să pro-

ducă cantitatea necesară de energie, să sintetizeze substanţele necesare şi să asambleze structurile cerute. Un alt efect este generarea produşilor finali.

Pentru fiecare dintre substanţe se poate găsi un lanţ „sursă — acumulare — orificiu de evacuare“. De aceea sistemul energetic al organismului este considerat adesea ca un set de surse, debite şi evacuări (Cooney, 1976).

1.3. *Analogie între organism şi maşină*

Schimbul şi producerea de substanţe într-un organism se realizează într-un fel, similar cu cel realizat în maşini complexe. De aceea din cele mai vechi timpuri s-a încetăţenit obiceiul de a trasa analogii între organisme şi maşini (tabelul 2).

În prezent, mai des întîlnită este analogia dintre un organism şi o instalaţie chimică.

În trecut analogia cu o instalaţie era de o importanţă pur teoretică. Acum, cînd sistemele de reglare constituite din elemente biologice şi tehnice au devenit realitate, această analogie a căpătat un anumit sens practic.

Astăzi, cea mai apropiată analogie cu un organism poate fi considerată o mare întreprindere, a cărei producţie este formată din unităţi chimice şi de asamblare, transportul fiind realizat de sisteme fiziologice iar conducerea, de un sistem cibernetic.

Tabelul 1

Funcţiile organismului ca sistem deschis
(Novoselţev, 1983, modificat)

Funcţie	Substanţe formate	Sisteme ale organismului
1. Primire de combustibil şi oxidant din exterior	*Substraturi:* lipide, vitamine, hidrocarburi *Combustibil:* hidrocarburi, lipide *Oxidant:* oxigen	tractul gastrointestinal, aparatul digestiv respiraţie, circulaţia sîngelui
2. Extragerea energiei din substanţe nutritive	ATP, căldură	sisteme biochimice, sisteme de termoreglare
3. Sinteza de substanţe şi formarea de structuri	*Sinteză de substraturi:* aminoacizi, proteine, acizi nucleici enzime *Formare de structuri:*	sisteme biochimice sistem endocrin sistem biochimic (structuri intracelulare)

Tabelul 1.

Funcţie	Substanţe formate	Sisteme ale organismului
4. Eliminare şi excreţie de produşi finali	*Eliminare:* molecule medii şi mari resturi necrozate	— sistemul reticulo-endotelial
	Excreţie: nitrogen, uree, creatină;	— rinichi, ficat
	apă, bioxid de carbon	— respiraţie, circulaţia sanguină
	bilirubină	— sistemul biliar (bila)

Tabelul 2

Analogii între organism şi maşină
(Colow, 1976, modificat)

Analogie	Autor	Prototip	Autor
Maşină mecanică	Descartes 1595—1650	Mecanică	Galileo 1654—1642
	Lamettrie 1709—1751		Newton 1642—1727
Maşină termică	Lavoisier 1743—1749	Termodinamică	Cornot 1796—1832
	Rubner 1854—1932		
Instalaţie chimică	Pasteur 1822—1895		
Instalaţie chimică cu reglare cibernetică	Biochimia şi cibernetica modernă	Cibernetică, Termodinamica sistemelor deschise	Wiener 1894—1967 Prigogine (1952)

1.4. *Sistemul cibernetic al organismului*

Sistemul cibernetic al organismului are mai multe nivele de reglare: genetic, biochimic, psihic şi comportamental. Genetic, mecanismele de reglare realizează procesele de asamblare şi reglarea biochimică a proceselor de sinteză. Mecanismele de reglare psihică furnizează transportul şi stocarea substanţelor. Răspunsurile comportamentale ale organismului, ca opinie generală, sînt menite să menţină activitatea mecanismelor de tip inferior şi să extindă gama condiţiilor pentru funcţionarea lor efectivă.

Scopul reglării în organism este să-i furnizeze supravieţuirea şi apărarea. De aceea, principala atribuţie a mecanismelor de reglare este să menţină nivelele apropiate ale proceselor vitale — de asamblare, sinteză şi transport. Totuşi istoriceşte, o altă problemă, care face parte din reglarea organismului, a devenit mai populară atît pentru specialişti cît şi pentru nespecialişti — aceea a menţinerii homeostazei.

Homeostaza este o constantă relativă a concentrațiilor unor substanțe în organism în condiții variabile de activitate și de mediu înconjurător (Bernard, 1980; Cannon, 1932).

Este de asemenea, bine cunoscut că toate organismele sînt bine adaptate mediului lor înconjurător, iar mecanismele lor operează economic și eficient. O analogie cu reglarea optimală a sistemelor pare foarte potrivită pentru a descrie această proprietate a biosistemelor (Rosen 1967, Rasherosky 1965). Cu toate acestea în prezent nu este totul clar în această analogie (Novoselțev, 1978).

Aceste aspecte ale activității de reglare ale organismului vorbesc în favoarea concluziei că în biosisteme reglarea are multe scopuri. Scopurile se interacționează ierarhic. După cîte cunosc autorii, primul care a reliefat aceste analogii a fost I. M. Secenov (1952, p. 527), care a scris: „cantitățile de substanțe minerale care intră și care ies sînt egale, și aceasta demonstrează cu siguranță cantitatea constantă existentă în organism".

De aceea, sarcina menținerii homeostazei într-un organism este un scop secundar, scopul principal fiind să mențină nivelele cerute de procesele vitale în asamblările structurilor, sintetizarea și transportul de substanțe.

Din tradiție, adevărata noțiune de reglare este tratată în biosisteme cu un sens mai larg decît în tehnologie și în teoria reglării în general.

Mecanismele de reglare includ și așa-numitele mecanisme pasive (încorporate). Acestea nu sînt structuri izolate în cadrul unui sistem, asemenea reglări pasive fiind un rezultat al interacțiunii elementelor întregului sistem (Bertalanffy, 1973).

Un exemplu de reglare pasivă este efectul concentrației substanței în două compartimente x_1 și x_2, cu viteza de curgere între ele de:

$$y=k(x_2-x_1) \quad (1)$$

Această tradiție se dovedește destul de fructuoasă din moment ce fenomene variate în reglarea fluxului de substanță în biosisteme, sînt rezultatul interacțiunii mecanismelor pasive și active, incluzînd pe acelea ale comportării homeostatice care vor fi discutate în continuare (a se vedea secțiunea 1.7).

1.5. *Descrierea proceselor de reglare în sisteme energetice*

Se poate prezenta diagrama căilor de transport și transformările prin care trece fiecare substanță participantă la metabolism. În cel mai simplu caz este chiar lanțul arătat în Figura 4.

Sursele și evacuările într-un asemenea lanț nu au drepturi egale în organism.

Una dintre ele este cu rol de conducător (master) cealaltă cu rol de condus (slave). Să considerăm, spre exemplu, un lanț de procese fiziologice de transport. În general unul din capetele unui astfel de lanț este cuprins în sistemul energetic, în timp ce celălalt este în afara lui. Procesul din sistem joacă rolul de master. Viteza sa nu este controlată de meca-

nisme fiziologice şi serveşte ca punct de referinţă pentru viteza procesului slave. Viteza procesului master determină viteza de curgere în întreg lanţul. Sistemul fiziologic furnizează conducerea procesului slave pentru a menţine balanţa de substanţe în organism.

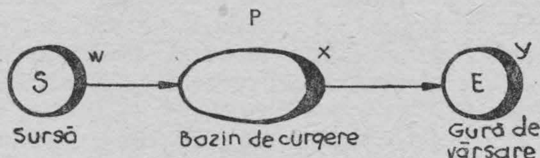


Figura 4. Circulaţia substanţei K în sistemul energetic: S — sursă; P — bazin de curgere; E — orificiu de evacuare.

De exemplu, pentru oxigen, absorbţia în procesele metabolice este master, iar sursa (procesul de respiraţie) este slave. Pentru bilirubină sursa este master (procesul descompunerii hemoglobinei şi generarea de bilirubină), iar absorbţia este slave (procesul de excreţie a bilirubinei cu bila).

În cazurile mai complexe lanţul din Fig. 4 poate avea mai multe surse şi puncte de absorbţie. De exemplu, aşa cum se arată în Fig. 3, asemenea lanţuri pentru energie au 2 puncte de absorbţie şi pentru produsul final două surse.

Vom nota viteza procesului master cu w_K, viteza procesului slave cu y_j, şi cantitatea de substanţă în rezervor cu x_i. Să considerăm că biosistemul are n_1 puncte de absorbţie primară (master) şi n_2 surse master,

$$n_1+n_2=n$$

Atunci vectorul:

$$w=[w_1\ w_2 \dots w_{n1}\ w_{n1+1} \dots w_n]^t \qquad (2)$$

determină necesarul uzual pentru organism în substanţe achiziţionate şi eliminate. Dacă m substanţe participă în procesul de transport şi schimb, pentru fiecare dintre ele vom putea scrie ecuaţia diferenţială ordinară

$$\dot{x}_i=\sum_{K=1}^{n} p_{ik}\ w_k+\sum_{j=1}^{n} q_{ij}y_j \qquad (3)$$

unde i=1, . . . , m şi p_{ik}, q_{in} iau valorile 0 sau ± 1.

Vitezele debitelor y_j în (3) sînt controlate în sistem şi anume:

$$y_i=F_j(x, v); \qquad j=1, \dots N \qquad (4)$$

unde:

$$x=[x_1\ x_2 \dots x_m]^t \qquad (5)$$

şi

$$v=[v_1\ v_2 \dots v_L]^t \qquad (6)$$

reprezintă vectorul de stare a sistemului și respectiv vectorul condițiilor de mediu. Funcțiile F_j în (4) descriu formarea fluxurilor secundare sub controlul mecanismelor active și pasive.

Ecuația de aproximări lineare (3) este transformată într-un sistem ordinar utilizînd ecuațiile de stare și de ieșiri

$$\dot{x}=Ax+Bv+Pw,$$
$$z=Cx+Dv \qquad (7)$$

Aici x este vectorul de stare al sistemului, v și w sînt vectorii de intrare și matricile A, B, P, C și D sînt de dimensiunile: (m×m), (m×L), (m×n), (M×m) și respectiv (M×L). De notat că într-o astfel de descriere vectorul de stare al biosistemului coincide cu cantitățile de substanțe în rezervorul sistemului. Vectorul de ieșire al sistemului Z este compus din vectorul vitezelor pentru fluxurile controlate (slave), y_j și toate semnalele de ieșire ale mecanismului de reglare activă u_r cu $r=N+1$, $N+2, \ldots M$:

$$Z=[y_1 \ldots y_N, u_{N+1} \ldots u_M]^t \qquad (8)$$

O diagramă bloc a proceselor de transport și schimb în sistemul energetic este arătată în fig. 5.

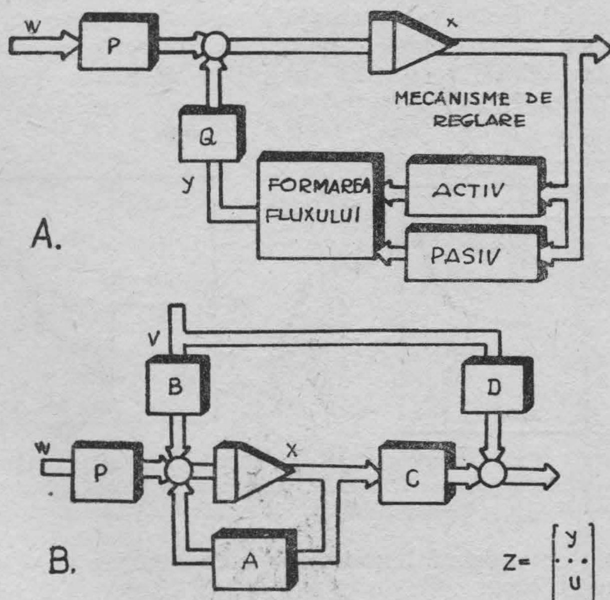


Figura 5. Schema bloc a sistemului de reglare pentru procesele de schimb și de transport într-un biosistem:
A: caz general; B: aproximația liniară.

1.6. *Satisfacerea nevoilor organismului*

După cum s-a notat mai sus scopul major al reglării în biosistem constă în satisfacerea necesităţilor organismului ca şi menţinerea stării stabile de neechilibru termodinamic.

Satisfacerea necesităţilor înseamnă că ecuaţiile (3) sau (7) au soluţii staţionare netriviale $x^s = x^s(V)$. Deoarece o viaţă normală este posibilă de obicei doar pentru anumite valori „normale“ x, $x \in \Omega_x$, valorile staţionare ale vectorului x^s trebuie să se găsească exact în această zonă. De aceea este necesar să exprimăm condiţiile de satisfacere a nevoilor organismului în felul următor:

$$\sum_{j=1}^{N} q_{ij} F_j(x^s, v) = \sum_{K=1}^{n} p_{ik} w_K \qquad (9)$$

$$x^s \in \Omega_x$$

$$i = 1, 2, \ldots, m.$$

Se poate observa din (1) că mecanismele interne (pasive) pot produce unele efecte asupra vitezelor fluxurilor de substanţe, deci componentele lui x joacă rolul de comandă în sistem.

1.7. *Homeostaza*

Celălalt scop al conducerii sistemului este să menţină homeostaza (vezi secţiunea 1.4). Relativa constanţă a mediului intern al organismului se poate trata ca o invariantă a anumitor componente a vectorului de stare x^s la schimbări ale vectorului condiţiilor de mediu v într-o anumită gamă dată (Novoselţev 1981).

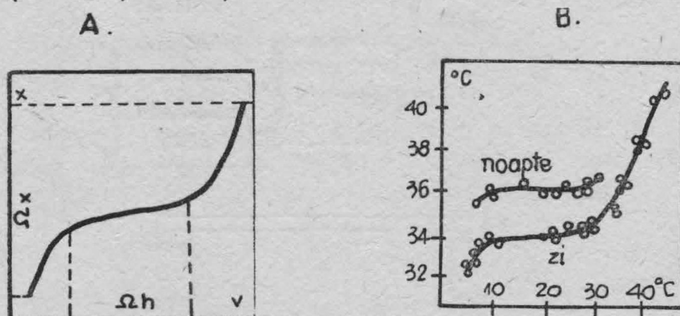


Figura 6. Curbă homeostatică. A: teoretică, B: experimentală: Ω_h — zona homeostatică; Ω_x — zona activităţii vitale (curba homeostatică a temperaturii corpului unui oposum, Prosser, 1973).

În sistemele ce exprimă homeostaza, se izolează o zonă referitoare la homeostază pentru care:

$$\tau_{il} \ll \tau^{\circ}, \qquad (10)$$

unde τ_{il} este valoarea absolută $\left|\frac{\partial x_i^s}{\partial v_l}\right|$ şi τ° este valoarea caracteristică τ_{il}

în afara zonei de homeostază. Relația τ_{ii} (v_i) în sisteme homeostatice este o curbă cu o porțiune liniară (platou) în zona de homeostază (Fig. 6).

Mecanismele active de reglare joacă rolul fundamental în menținerea homeostazei. În zona de homeostază reglarea este realizată aproape exclusiv prin aceste mecanisme astfel încît variabilele de reglare ale mecanismului pasiv x^s rămîn practic invariante la schimbările lui v.

După ce se consumă resursele mecanismului activ sistemul trece de porțiunea liniară a curbei și starea staționară (8) este menținută în continuare de mecanismele pasive.

Exemplu: Considerăm o diagramă simplificată a transportului de oxigen în organism, atunci cînd cantitatea de oxigen în aerul inhalat este redusă. În exemplu sînt prezente două mecanisme de reglare: cel pasiv este menit să crească extragerea de oxigen dintr-o unitate de volum a sîngelui, în acord cu relația (1), și cel activ să îmbunătățească fluxul sangvin. Pentru scopuri didactice ne restrîngem la descrierea cea mai simplă a acestor mecanisme. Blocurile din Fig. 5 A sînt descrise de ecuațiile scalare de forma (11) și (12).

Procesul de stocare a oxigenului și de consum al lui este:

$$\dot{x} = \frac{1}{V}(y - w), \tag{11}$$

unde V este o constantă.

Viteza controlată de intrare a oxigenului este

$$y = \alpha Q, \tag{12}$$

unde mecanismele care controlează variabilele α și Q sînt descrise de:

$$\alpha = k_p(w - x)$$

$$Q = \begin{cases} Q_M, & k_a(\bar{x} - x) \geqslant Q_M \\ k_a(\bar{x} - x), & Q_M > k_a(\bar{x} - x) \geqslant Q_M \\ Qm, & k_a(\bar{x} - x) < Qm \end{cases} \tag{13}$$

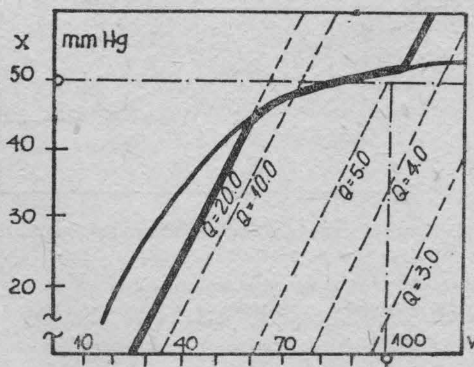


Figura 7. Curba homeostatică în sistemul de transport al oxigenului.

Cerințele organismului sînt satisfăcute dacă în concordanță cu (9), este valabilă ecuația

$$y(x^s, v) = w \tag{14}$$

Aici Q este fluxul sangvin, l/min; α este cantitatea de O_2 ce intră în organism într-un litru de sînge, ml/l; kp și ka sînt cîștigurile mecanismelor pasive și respectiv active. Să considerăm:

$k_p=1{,}0$, $k_a=2{,}0$, $\bar{x}=52{,}5$ (mmHg), Qm=5,0 și $Q_M=20{,}0$. Atunci în condiții normale cu w=250 ml/min și v=100,0 mm Hg se găsește: $\bar{x}^s=50{,}0$, Q=5,0 și $\alpha=50{,}0$.

Fig. 7 trasează grafic relația: $\bar{x}^s(v)$ pentru care (14) este satisfăcută. Linia neîntreruptă corespunde lui (13) în timp ce linia punctată arată relația cu Q=const. Pentru aceste curbe $\tau°=1{,}0$. Forma finală a curbei este indicată de linia îngroșată.

Extinzînd regiunea liniară corespunzător saturației mecanismelor active se controlează viteza de intrare a oxigenului. În regiunea liniară după cum se deduce din (10),

$$\left|\frac{\partial x^s(v)}{\partial v}\right| \ll \tau°.$$

1.8. *Conceptul de valoare nominală de referință (punct fixat)*

Dacă cîștigurile mecanismelor de comandă active și pasive sînt în relația $k_a \gg k_p$, punctul de activare a mecanismelor active ($\bar{x}$ în exemplul

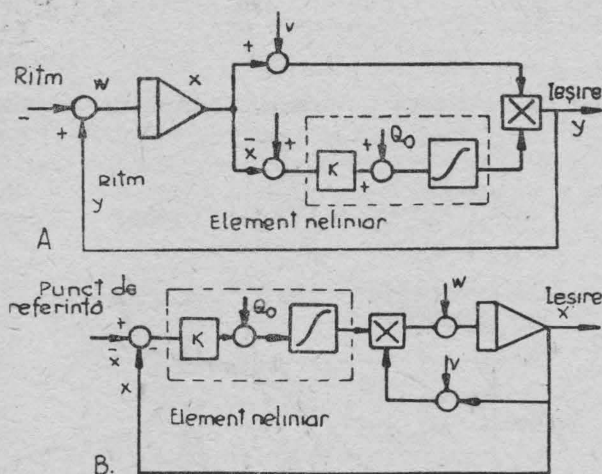


Figura 8. Două forme de reprezentare a fluxului pentru reglarea alimentării cu oxigen:

A — model sistem comportamental deschis cu $\bar{x}$ ca o valoare de prag; B — model cibernetic al homeostazei cu $\bar{x}$ ca o valoare de referință.

din secțiunea 1.7) se poate trata cu bună aproximație ca valoare de referință pentru mărimea x.

Acest model a fost cel mai adesea discutat în literatură pentru modelări de biosisteme. Așa-numitul „model cibernetic al homeostazei" este un sistem de reglare obișnuit, cu valoare de referință (Grodins, 1963; Milsum, 1966; Ray, 1974).

Diferența în a descrie biosistemele ca sisteme deschise (prin faptul că furnizează schimburile de substanțe energetice) sau ca sisteme de reglare cu valoare de referință, nu este de fapt așa critică. Același model al unui biosistem poate fi prezentat în oricare din cele două forme.

În Fig. 8 se prezintă o diagramă de flux pentru sistemul de aprovizionare cu oxigen, considerat în secțiunea 1.7, mai întîi ca un model de sistem comportamental deschis și apoi ca un model cibernetic al homeostazei pentru variabila **x**.

Cele două modele trasează o structură practic identică.

1.9. *Discuție*

Reglarea pentru biosisteme cunoaște astăzi o atenție crescută din partea specialiștilor din domeniul teoriei reglării. Motivul este determinat de două aspecte.

Mai întîi, multe probleme practice cer ca organismul să fie tratat ca o instalație reglată. Acestea sînt, spre exemplificare, problemele reglării organelor interne artificiale. Progresul tehnic este astăzi în mod evident la nivelul schimbului de substanțe și energie între organism și mediul său înconjurător.

De aceea atenția principală a lucrării de față s-a îndreptat spre sistemele energetice ale organismului. În afara scopului lucrării, sînt cîteva zone ale reglării în biosisteme la fel de importante. Acestea includ problema obținerii și prelucrării de informație (modelarea analizatorilor, senzorilor și receptorilor) și problema investigării mecanismelor de acționare ale reglării (adică comanda activității de mișcare).

Cel de-al doilea motiv, dar nu de o importanță mai mică, al aplicării active a tehnicilor teoriei reglării pentru descrierea reglării biosistemelor este acela că aceste tehnici permit o privire mai adîncă în esența multor fenomene vitale. Ne putem chiar aștepta că progresul în implementarea mijloacelor tehnice ce influențează organismul și în obținerea unei mai mari experiențe a existenței organismului într-un mediu înconjurător devenit avansat din punct de vedere tehnic, va duce la anumite corecții și revizuiri a conceptelor de bază referitoare la viață în general (Novoselțev, 1982).

2. Reglarea pentru biosisteme

Sistemele vii și mașinile lucrează împreună de foarte mult timp. Mijloacele tehnice au început să fie folosite de mulți ani ca o prelungire pentru organism. La început ele erau simple îmbrăcăminți ce permiteau oamenilor să supraviețuiască într-un climat aspru. Apoi au apărut membre mecanice, artificiale, ca analogii inerte a pielii și oaselor. Sistemele de receptori ai organismului au cîștigat sticla optică și dispozitive vizuale și auditive de complexitate ridicată.

În caz de afecțiuni, chirurgia reparatorie restabilește integritatea laringelui, a vaselor de sînge a oaselor și ligamentelor. Mijloacele tehnice la

un anumit nivel au devenit atît de complicate, încît implementarea lor eficientă necesită proiectarea unor sisteme de reglare speciale. Împreună cu aviaţia şi tehnologia spaţială şi electronică, instrumentaţia medicală a căpătat o dezvoltare rapidă promiţînd să devină o ramură industrială (Viktorov, 1981; Akhutin, 1981; Nalecz, 1981).

Exemplele reglării funcţiilor organismului atît pentru oameni cît şi pentru animale pot fi găsite la diferite nivele — social, genetic, psihologic şi biochimic. Cel mai adesea pentru o problemă practică, toate aceste nivele sînt inseparabile. De aceea pentru a asigura o putere de lucru sporită necesară piloţilor şi cosmonauţilor (27 pînă la 30 ore de zbor) se folosesc următoarele mijloace şi tehnici de control şi reglare:

— programe individuale ale perioadei de muncă şi odihnă;

— mijloace fiziologice şi igienice (incluzînd acelea menite să menţină parametri vitali);

— mijloace şi tehnici pentru igiena psihică (autoreglarea psihosomatică;

— mijloace electrofiziologioe (simularea electrică a sistemului neuromuscular, acţiunile asupra punctelor active ale organismului);

— preparate farmaceutice (tranchilizante, stimulatori pentru sistemul central nervos);

— mijloace fizice (procedeuri hidro, saună, masaje);

— autoantrenarea şi autosugestia.

Realizările practice ale reglării funcţiilor externe ale organismului cer proiectarea unei largi clase de mijloace tehnice, operînd împreună cu obiecte biologice.

În ultimii zece ani aceste legături între complexele tehnice şi biologice s-au dezvoltat extensiv.

2.1. *Complexe biotehnice*

Societatea modernă este adesea intitulată societate industrială, ceea ce subliniază importanţa componentelor tehnice din structura sa.

Mijloacele tehnice create de om au pătruns în toate sferele activităţii sale.

Integritatea obiectelor de natură tehnică şi biologică, stabilită de relaţiile lor unilaterale, sau mutuale, pentru un anumit interval de timp, sau pentru obţinerea unui anumit scop, vor fi în continuare referite ca un complex biotehnic (CBT).

Rolul componentelor tehnice într-un asemenea complex pot fi estimate convenţional printr-o scară de proporţii, în care o mare cantitate de mijloace tehnice din CBT sînt asociate cu îmbunătăţirea potenţialului lor funcţional. Această creştere potenţială de la funcţiile de observaţie la cele de comandă şi reglare poate fi exprimată de schema informaţie-ghidare-acţiune.

Această clasificare simplistă a CBT (Fig. 9) poate fi folosită pentru descrierea nivelului de astăzi al avansului în „industrializarea" conducerii pentru biosisteme.

Continuînd avansul atins în științele medicobiologice, CBT furnizează astăzi terenul necesar progresului în cele mai importante arii ale serviciului medical — diagnosticarea, terapeutica și alte activități (Viktorov, 1979; Viktorov, Kerbunov, 1979).

2.2. *CBT pentru diagnosticare*

CBT pentru diagnosticare sînt caracterizate printr-o relație unilaterală între biologic și tehnic și anume de la primul spre ultimul. (Viktorov, Dubrovsky, 1982.)

În lume există mii de instrumente diverse cu o oarecare legătură directă cu pacientul formînd CBT pentru diagnosticare (Fig. 10).

Dezvoltarea mijloacelor de diagnosticare a cunoscut în ultimul timp o ascensiune rapidă, cînd medicii și-au dat seama de necesitatea unei interpretări complexe a problemelor de diagnosticare. Obținerea unui mare

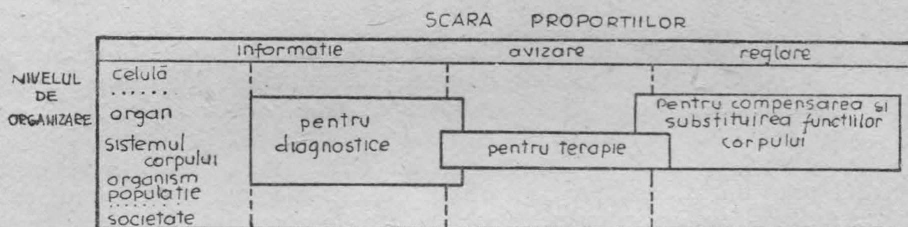


Figura 9. Clasificarea complexelor biotehnice.

număr de informații cere mijloace tehnice diverse făcînd adesea procedurile de diagnosticare nereale.

De aceea a fost necesară proiectarea mijloacelor tehnice capabile să producă estimări asupra mai multor caracteristici majore sau indici.

Informațiile pentru obținerea parametrilor diagnosticului în CBT modern sînt date de:

— radiațiile electromagnetice ale organelor și corpului omului acoperind toată gama spectrului;
— semnale video;
— semnale audio;
— radiațiile inimii;
— interacțiuni chimice (teste);
— interacțiuni mecanice (teste);
— interacțiuni psihofarmacologice.

Cele mai eficiente mijloace de diagnosticare sînt aparatele de prelevare de tip electrofiziologic. De aceea cele mai dezvoltate sînt radiațiile electromagnetice folosite de CBT ca sursă de informație asupra stării organismului.

Aplicarea calculatoarelor, ca parte a diagnosticelor CBT, permit să fie rezolvate o întreagă gamă de probleme ce furnizează viteză și vizualizarea fiabilă a obiectelor în timpul studiului.

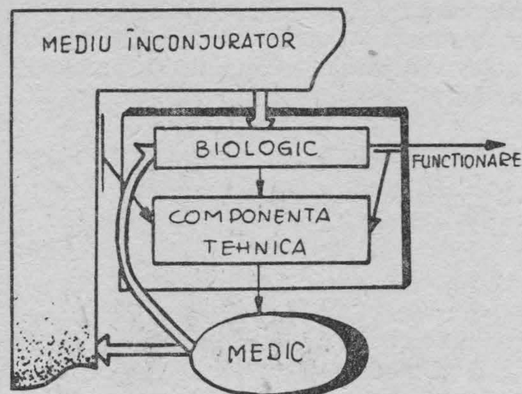


Figura 10. Complex biotehnic pentru diagnosticare.

Trebuie menţionate aici şi metodele şi mijloacele detecţiei cu ultrasunete şi a controlului organelor interne şi ţesuturilor. Menţinînd trăsăturile pozitive ale tehnicilor de radioscopie tradiţională cu raze X, ele sînt libere de efectele negative ale expunerii la radiaţii.

Componenta tehnică a unui asemenea CBT este construită în jurul efectului unei surse/receptor de radiaţii electro-magnetice şi a unui calculator (adesea cu un microprocesor) folosit pentru reconstituirea organului şi afişarea sa în timpul studiului.

Sistemul de investigare tomografic cu raze gama constituie o clasă promiţătoare a CBT. Aceste sisteme încorporează de obicei calculatoare de capacitate medie (Viktorov, 1981).

În prezent pentru practica medicală este folosit un CBT — utilizînd principiul rezonanţei magneto-nucleare (RMN). În cele mai multe diagnosticări ale CBT componentele biologice afectează pe cele tehnice într-un mod pasiv. Totuşi pot fi date unele exemple ale proiectării de CBT în care componentele semnalului biologic nu sînt doar transmise în cele tehnice, sau captate de acestea pentru o vizualizare ulterioară, ci sînt mai ales folosite pentru a influenţa activ modul de operare a ultimelor.

O realizare tehnică a timpului nostru a fost crearea unui endoscop flexibil care stă la baza CBT ce foloseşte un semnal video ca sursă de informaţie. Întreaga natură a folosirii sale a impus cerinţe stringente pentru proiectarea sa, şi proprietăţi tehnico-medicale, dînd un exemplu sugestiv asupra modului în care originea biologică influenţează soluţia tehnică. Au fost proiectate diverse tipuri de endoscoape ce furnizează medicilor o nouă tehnică pentru investigaţiile medicale ale cavităţilor interne ale organismului, fără necesitatea unei intervenţii chirurgicale.

2.3. *CBT pentru terapeutică*

Marea importanţă a echipamentelor pentru CBT şi creşterea potenţialului lor funcţional, permit aplicarea acestora nu numai la diagnosticare dar şi obţinerea unui ghid de tip „avertisment“ pentru medic (Fig. 11).

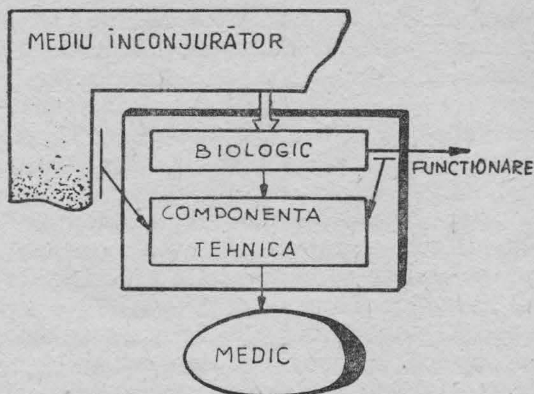


Figura 11. Complex biotehnic pentru terapie.

În acest caz informaţia bio-medicală asupra pacientului se încheie în cele din urmă prin a lua forma de „concept de diagnosticare" care reprezintă pentru medic un subiect de verificare. Acest mod semiautomat de operare serveşte, de exemplu, în sistemele asistate de calculator pentru analizele semnalelor electrice ale cordului (Caceres, 1974). În acest domeniu sînt descrise anumite scheme algoritmice care dau diagnosticări ale patologiei cardiace (Pipberger, 1975). Avansul în acest domeniu este caracterizat de faptul că o analiză comparativă a interpretărilor algoritmului ECG prin calculator şi făcută de specialişti, vorbeşte în favoarea tehnicilor de calcul care sînt mai puţin expuse erorilor subiective.

Un alt exemplu evident al folosirii efective a CBT — aşa-numitul sistem de preselecţie — este sistemul folosit pentru examinarea preventivă a maselor de populaţie. Componenta biologică a unor asemenea sisteme este un număr de grupe de pacienţi, în concordanţă cu un anumit index (zonă de locuit, profesie, vîrstă etc.).

Componenta biologică „extinsă" a CBT se adaugă dezvoltării componentelor tehnice multifuncţionale — un sistem de mijloace de diagnosticare medicală multiplă (mijloace pentru măsurări cardiace, spirometrice, analize gastroenterologice, mamografii etc.).

Operaţiile CBT de preselecţie în masă intenţionează să pregătească „macrodecizii", alternative pentru tratamentul în viitor al fiecărui pacient.

Participarea unui medic la oricare din studiile procesului dă flexibilitate sistemului, ceea ce răspunde scopului şi complexităţii problemei. Experienţa menţinerii extensive a unor asemenea CBT este descrisă în literatură (Kiraki şi ceilalţi, 1977; Kevin O'Kane, Edward, 1977).

Sînt larg utilizate în practica medicală CBT folosite în supravegherea pacienţilor aflaţi în stări grele sau neobişnuite (perioade postoperatorii, şocuri etc.). Aceste CBT sînt uneori denumite staţii de monitorizare. Ele diferă în complexitate şi potenţial funcţional. Aceasta permite să se diferenţieze între ele două tipuri de complexe, un tip pentru ghid (sfătuitor) şi altul ce asigură premisele reglării automate a componentelor biologice încorporate în organismul pacientului.

Primul grup include sisteme bine cunoscute de indicare a parametrilor vitali, sisteme de alarmă, sisteme de predicţie a stărilor etc. „Închiderea" buclei pentru aceste sisteme este realizată de acţiuni ale personalului medical.

Astfel, reglarea respiraţiei în CBT este realizată cu semnale electrice pulmonare sau probe, ale respiraţiei nazale. La pierderea respiraţiei se declanşează un semnal de avertizare.

Un exemplu bun, al unui alt CBT reglat biologic, este un complex automat pentru menţinerea duratei necesare a anesteziei. Un alt exemplu de bioreglare este un aparat pentru ventilaţia artificială a plămînului cu intensitate reglată în mod automat.

Bioingineria este înglobată în proiectarea şi implementarea unui CBT pentru terapia intraoptică. În insuficienţele cardiace acute (infarcturi miocardice, chirurgie pe cord-deschis) sistemul de circulaţie al sîngelui pacientului este conectat la aparatura circulatorie extracorporală.

2.4. *CBT pentru compensarea sau substituirea anumitor funcţiuni*

Cel mai tipic reprezentant al unui asemenea CBT este un dispozitiv pentru substituirea temporară a funcţiei de excreţie a rinichilor.

Eficienţa clinică a complexului capabil de o asemenea substituţie, dovedeşte în principal, aplicabilitatea mijloacelor tehnice pentru succesul susţinerii pe timp îndelungat a vieţii biologice, incluzînd organismul uman.

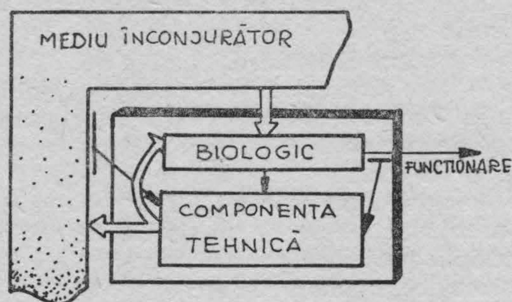


Figura 12. Complex biotehnic pentru compensarea sau înlocuirea funcţiilor corpului viu.

Exercitînd funcţia unui regulator pentru starea obiectelor biologice, CBT furnizează stabilitatea concentraţiei ionice, înlăturarea surplusului de apă şi curăţirea de produse toxice cu greutate moleculară mică sau medie.

Funcţionarea CBT se întrerupe pe timpul folosirii mediului regulator în interacţiune directă prin membrane (naturale sau artificiale) cu mediul reglat (plasmă, sînge, limfă).

Mediile regulatoare se împart în naturale şi artificiale (Khaithin, 1980).

Se folosesc ca medii de reglare artificiale soluții absorbante și special preparate avînd o concentrație ionică normală fiziologic.

Mediul de reglare natural este mutat ultrafiltrat din mediul reglat sau de la sîngele donatorilor.

CBT include aparate de transfer al maselor (dializatoare, filtre fine și coloane de absorbție), pompe de perfuzie, generatoare și distribuitori de soluții regulatoare, elemente de comunicație (conducte, punți, șunturi și fistule) precum și mijloace de comandă și reglare.

Proiectarea CBT moderne asistate de calculator care utilizează elemente cu o unică folosire pentru contactul direct cu corpul uman a îmbunătățit esențial reproductibilitatea efectelor de reglare și, în același timp, a simplificat reglarea și service-ul instrumentației și echipamentelor. Astfel s-au eliminat practic complicațiile asociate trăsăturilor constructive ale CBT și încălcarea modului de operație stabilit de operator.

Totuși este încă urgentă sarcina eliminării disproporțiilor dintre nivelele înalte de automatizare și nivelul insuficient al achiziției și prelucrării datelor, care să furnizeze operatorului informația necesară pentru a lua deciziile referitoare la modul de funcționare a CBT, și pentru reglarea directă.

Introducerea achiziției de date și aparaturii de prelucrare în echipamentele CBT care furnizează informația necesară în mare parte pentru a duce la o reglare suficientă și pentru a elimina redundanța informațiilor, va permite îmbunătățirea performanțelor de siguranță ale CBT. Aceasta se datorește optimizării modului său de operare și reducerii probabilității introducerii de către operator a unui mod de lucru anormal fiziologic.

Primele rezultate au fost obținute în această direcție, de exemplu, în proiectarea aparaturii „rinichiului artificial" AIP-A-01 (U.R.S.S.), sau „Citatron TM" de către Cordis Dow (Savchenko și alții, 1980).

Un număr de CBT sînt proiectate să realizeze bioreglarea simulării artificiale a organelor corpului. Acestea sînt, mai întîi de toate, aparaturile ce generează pulsuri electrice pentru mușchii excitatori cardiaci, simulatoarele cardiace etc. O dată cu dezvoltarea proiectării tehnicilor de electrosimulare, acestea au pătruns în sfera reglării aparatului locomotor uman. Un astfel de CBT include aparate pentru prelevări biologice, un amplificator și un convertor de semnal, mijloace de comunicație și transfer a semnalului captat de la componentele tehnice spre cele biologice. Se poate sublinia marea capacitate de refacere a acestor CBT (gimnastică de recuperare pentru organele paralizate). Începînd cu mijlocul deceniului șase, cercetătorii au introdus și au dezvoltat aplicațiile așa-numitei tehnici de reacție biologică (biofeedback). Scopul CBT, derivînd din folosirea acestei tehnici, este de a realiza procesul de recuperare a pacienților cu afecțiuni ale sistemului motor, rezultate din leziunile sistemului nervos. Canalele senzoriale (de exemplu, canalul optic) sînt folosite pentru a introduce un semnal de reacție generat la unele imagini ale activității mioelectrice ale pacientului. Observarea acestei imagini permite pacientului să-și corecteze conștiincios mișcarea, crescînd puterea parezei mușchilor și reducînd tonusul grupelor musculare spastice (Leuner, 1977).

Aceeași metodă de bază este utilizată pentru dezvoltarea tehnicilor de tratament pentru un număr de tulburări psihosomatice prin învățarea pacienților, dînd apoi informații asupra stării fiziologice a organismului (tensiune arterială, temperatura corpului, vîrfurile encefalogramei individuale etc.).

Reacția biologică este folosită activ în CBT cu intenția de îmbunătățire a condițiilor de viață a pacienților orbi. Proiectarea mijloacelor pentru orientarea spațială a orbilor, deși folosește tehnici de localizare foarte bine cunoscute (ultrasunete, laser etc.) au lăsat pentru un lung interval de timp o impresie de neconcordanță între componentele tehnice și biologice. Problema compatibilității lor se menține ca o cheie a rezolvării finale a acestui obiectiv.

2.5. *Aspecte informatice ale funcției de reglare a organismului*

Din punct de vedere al reglării, informația despre biosisteme poate fi apriorică sau curentă. Informația apriorică este stocată cu ajutorul disciplinelor biologice și fiziologice speciale. Informația curentă de la biosisteme se obține direct în timpul reglării procesului, într-o formă obișnuită pentru sarcinile reglării.

O trăsătură specifică a problemelor de reglare pentru biosisteme este aceea că prezintă o mare cantitate de date apriorice, cantitatea de informație ce vine de la biosistemele reglate fiind relativ mică. Această informație este de obicei insuficientă pentru organizarea unei scheme de reglare, similare celei din sistemele tehnice.

Pot fi menționate aici două tendințe. Una este asociată cu o creștere directă a datelor de intrare. Pentru acest scop sînt proiectate și implementate noi mijloace tehnice pentru obținerea informației de la obiectele biologice. Trăsăturile principale ale acestei tendințe sînt prezentate în secțiunile 2.2. pînă la 2.4.

O altă tendință este îndreptată spre o folosire mai eficace a informației apriorice și a informației în timp real. Trăsăturile ei principale sînt măsurătorile bazate pe un model. Așa-numitele măsurători deductibile ale sistemelor fiziologice, au devenit o cale promițătoare și realizabilă pentru obținerea datelor asupra valorilor variabilelor interne și parametrilor neaccesibili măsurătorilor directe (Carson, Finkelstein, 1981).

Selecția unor modele matematice adecvate joacă un rol esențial în măsurătorile deductibile. Aplicarea modelelor în fiziologie și la îngrijirea pacientului constituie astăzi mai mult sau mai puțin un standard adecvat. (Carson, Cobelli, Finkelstein 1982). Tehnicile pentru proiectarea unor asemenea modele devin tot mai formalizate. Aceste metodologii speciale se dezvoltă în domeniul formulării, identificării și validării unor asemenea modele.

Aplicarea în creștere a tehnicilor de modelare matematică pentru reglarea funcțiunilor organismului înseamnă că o cantitate de informație curentă, insuficientă pentru procesul reglat poate, într-o anumită dezvol-

tare, să fie completată și pe baza informațiilor apriorice. În aceste cazuri devin mai vitale metodele descrierii formalizate a proceselor reglate în organismele vii, tratate în prima parte a lucrării (secțiunile 1.2. la 1.5.).

2.6. *Fiziologia inginerească*

Reglarea funcționării organismului este orientată în mod normal în sensul unei recuperări în sistemul energetic al procesului, care poate fi prezentat ca un set de surse, fluxuri și orificii de evacuare (Fig. 3). Funcționarea deficitară a unui asemenea sistem semnifică nerespectarea condițiilor (9) sau (10). Funcționarea normală se poate restabili în două moduri:

1. Corectînd viteza fluxurilor în sistemul existent fără a schimba vreo structură a sa. Un exemplu este protecția pasivă a căldurii în condiții de frig. Hainele reduc viteza pierderilor de căldură pînă la un nivel acceptabil. Un alt exemplu este stimulatorul cardiac ce menține circulația sîngelui pentru cordul natural (totuși bolnav), astfel încît să fie satisfăcute necesitățile curente ale organismului.

2. Modificînd structura dispeceratului fluxurilor prin introducerea unor noi surse sau orificii de evacuare artificiale. Un exemplu poate fi constituit de încălzirea activă a corpului unui scufundător sub apă (sistemul COMEX-PRO — vezi Griselin, 1982).

Un ventricul artificial al cordului servește ca sursă pentru sînge. Un rinichi artificial este un exemplu de orificiu de evacuare artificială. În amîndouă cazurile funcționarea normală se poate restabili prin realizarea unor CBT speciale. Să considerăm un CBT în care sînt generate sursele și orificiile de evacuare artificiale la nivelul sistemelor fiziologice ale organismului. Ele angajează dispozitive mecanice, chimice și electrice, corectînd funcționări variate în sistemul energetic și jucînd rolul surselor comandate și de orificii de evacuare în cadrul sistemului (vezi secțiunea 1.5).

Crearea mijloacelor tehnice pentru operațiile combinate cu sistemele fiziologice ale organismului, trebuie să se bazeze pe experiența și cunoștințele atît tehnice cît și biologice, fiziologice și ale științelor medicale.

Asemenea mijloace tehnice sînt într-un fel specifice. În medicină ele sînt numite organe interne artificiale. În cadrul secțiunii dedicate organelor artificiale interne, programul acestui Congres include secțiunea reglării glandelor endocrine. În cercetările bio-medicale (spațiu, mediu subacvatic) asemenea dispozitive sînt părți ale unităților de protecție individuale.

O diversitate de mijloace tehnice implicînd analizele lor și tehnici de descriere, devin din ce în ce mai uniforme. O ramură de cercetare rezultînd din aceste sinteze a fost intitulată fiziologie inginerească (Trapeznikov, 1981, Novoselțev, 1983 a).

Fiziologia inginerească studiază procese în sistemele fiziologice, ce operează în CBT în legătură cu mijloacele tehnice. Întreaga caracteristică specifică organismelor vii este prezentă în sistemele de fiziologie ingine-

rească. Oricum, ele sînt capabile de implementări ale reglării tipice pentru sistemele create de om. De aceea descrierea proceselor de reglare în sistemele fiziologice inginerești face un larg apel la metodele și tehnicile teoriei reglării automate.

Mijloacele tehnice folosite pentru îmbinarea lucrului cu sistemele fiziologice prin intermediul CBT sînt însumate în tabelul 3. Tabelul este un adaos fiziologico-ingineresc la listarea funcțiilor naturale ale organismului date anterior în tabelul 1.

Discuții

Astăzi, o tendință de perspectivă în reglarea biosistemelor constă în proiectarea unor mijloace tehnice eficiente pentru corelarea funcționării acestora cu sisteme ale organismului.

Aceasta stimulează pe de o parte dezvoltarea complexelor biotehnice simple pentru observarea și reglarea stării organismului și pe de altă parte, dezvoltarea unor noi metode teoretice de reglare a biosistemelor și realizări în teoria modelării.

Pe ansamblu, mai promițătoare este proiectarea și reglarea mediului artificial al omului.

Reglarea mediului de viață poate lua forma exotică a organelor „interne" ale organismului și în acest caz se poate localiza spațial în afară sau în interiorul organismului.

Adesea un organism care nu mai este capabil de o viață normală într-un mediu înconjurător natural, își poate păstra totuși o viață activă într-un mediu organizat artificial. Viața în asemenea medii artificiale a devenit un fenomen de masă. În jurul anului 2000 sute de mii de oameni vor trăi cu stimulatoare cardiace implementate, rinichi artificiali sau pancrease artificiale.

Oricum, căutările pentru depistarea căilor care să asigure necesitățile omenirii în resurse minerale și energie, amplifică importanța mijloacelor tehnice menite să asigure activitatea unui om sănătos în condiții de muncă, în noi medii de viață — în ocean, pe platformele continentale sau în spațiul cosmic din vecinătatea pămîntului. Acestea sînt domeniile fundamentale ale cercetărilor viitoare și ale dezvoltării viitoare în științele conducerii și reglării biosistemelor.

Tabelul 3

Mijloace tehnice folosite pentru corectarea funcțiilor organismului
(Novoselțev (1983 a) modificat)

Funcție	Substanțe	Mijloace tehnice
1. Primire din exterior	substraturi	sisteme de hrănire artificială
	combustibil	la fel
	oxidant	aparatură de respirație artificială
	substrat, combustibil și oxidant	aparat de oxigenerare hiperbarică

Tabelul 3

Funcție	Substanțe	Mijloace tehnice
		circulație sanguină auxiliară, cord artificial, aparatură cardiacă auxiliară (stimulator cardiac, aparate de masaj)
2. Extracția energiei	ATP	nu
	alte tipuri de energie (electrică, mecanică)	sisteme de transfer de putere transcutanate și percutanate, surse de puteri implantabile*
	energie calorică	sisteme de protejare a căldurii active pentru operatori și scafandri, sisteme auxiliare chirurgicale de răcire
3. Sinteze de substanțe și formare de structuri	substanțe active (insulină, droguri)	sisteme de dozare automată
	structuri specializate ale organismului	nu**
4. Eliminare și excreție de produși finali	eliminări în organism	nu
	excreția de substanțe în afara organismului: macromolecule	aparatură pentru hemosorbție și hemoperfuzie, rinichi artificial
	apă, acid carbonic	instalații respiratorii auxiliare

* Aceste forme de energie sînt consumate numai de organele artificiale (v. pag. 88).

** Substituirea completă a unui organ natural cu unul artificial sau transplantat, nu este privită aici ca o formare de structură naturală.

3. Cordul integral artificial — problemă tipică de reglare a organismului

3.1. *Cordul integral artificial*

Scopul creării unui cord integral artificial (CIA) constă în înlocuirea temporară sau permanentă a cordului bolnav cu o pompă. Din punct de vedere tehnic cordul natural reprezintă o pompă hidraulică cu ieșire variabilă și care menține circulația sîngelui prin organism în concordanță cu necesarul fiziologic al acestuia.

Din această perspectivă CIA este compus dintr-un dispozitiv de pompare cu două încăperi lucrînd pulsatoriu și menținînd transferul unidirecțional al sîngelui în circulația mare și mică, sistemul de acționare al pompei, sursele de energie și sistemul de reglare cu senzori incluzînd senzori bioinformaționali.

Proiectarea fiecăreia dintre părțile CIA menționate mai sus necesită rezolvarea unor probleme tehnice și medicale complexe atît de natură teoretică, cît și practică. În ceea ce privește sistemul de acționare a CIA, acesta trebuie să fie un dispozitiv cu masă și dimensiuni specifice pe unitatea de putere, cu unul sau două ordine de mărime mai mic decît în proiectarea acționărilor convenționale, ceea ce conferă o fiabilitate crescută și o durată de viață sporită.

Cît despre sursele de energie, trebuie proiectate surse cît mai mici dimensional dar de capacitate mare și care să permită o funcționare de lungă durată în interiorul corpului. Sînt necesare materiale cu proprietăți antitrombice, capabile să stea în contact de lungă durată cu sîngele și țesuturile corpului, fără a le afecta proprietățile. În proiectarea sistemului de reglare trebuie studiate amănunțit mecanismele fine ale reglării cordului natural și realizate soluții tehnice care să implementeze mecanismele de reglare cunoscute.

Complexitatea și cantitatea mare de activitate teoretică și experimentală necesare pentru proiectare CIA impune o realizare treptată a obiectivului propus. Unul dintre pașii inițiali îl reprezintă proiectarea unei versiuni de CIA cu acționare externă, în care se implantează numai dispozitivul de pompare, în timp ce restul elementelor sînt plasate în afara corpului. La acest pas trebuie îndeplinite următoarele sarcini: proiectarea unui dispozitiv de pompare implantabil și dezvoltarea algoritmilor de reglare care să asigure o fiabilitate suficientă precum și reproducerea integrală a funcțiilor de pompare ale cordului natural.

În primii ani de dezvoltare a proiectării CIA s-a acordat puțină atenție reglării automate. Motivul a constat în faptul că s-au abordat pentru început probleme privind crearea proiectelor de CIA optimal, căutarea de materiale hemocompatibile și proiectarea de acționări sigure, reduse dimensional.

În prezent, aproape toate centrele de cercetări din lume, angajate în proiectarea CIA se ocupă cu proiectarea și dezvoltarea sistemelor de reglare automată a CIA.

Primele lucrări în domeniul sistemelor de reglare a CIA au fost elaborate în cadrul universității Uta, Salt Lake City (S.U.A.), fiind conduse de către dr. W. Kolff. Acestea au fost urmate de rapoarte din alte centre de cercetare americane angajate în domeniu: Institutul de cardiologie Texas din Houston, condus de dr. T. Akutsu și Clinica Cleveland, condusă de dr. Y. Nose.

În U.R.S.S. activitatea în această direcție a început în 1971 în Institutul Unional de Cercetări pentru Chirurgie Clinică și Experimentală.

Mai nou, au apărut articole privind proiectarea sistemului de reglare a CIA în Berlinul de Vest, R.F.G., Franța, Japonia, Cehoslovacia și Italia.

3.2. *Principalele tendințe în sistemele de reglare a CIA*

În prezent se constată următoarele patru tendințe clare în sistemele de reglare automată a CIA:

1. Proiectarea sistemelor de reglare lucrînd cu volumul limitat de date specifice iniţiale, accesibile cercetătorului. În acest caz, în mod uzual, sursa de informaţie este un senzor pentru una din variabilele cordului artificial. Deoarece asemenea senzor este un element din sistemul de reglare în buclă închisă a CIA (fie controlat de un operator, fie complet automat), informaţia iniţială determină covîrşitor principiile care stau la baza proiectării sistemelor de reglare.

În particular se utilizează un senzor de poziţie capacitiv al membranei ventriculare artificiale. Scopul său de bază este de a furniza informaţii primare referitoare la ieşirea pompei sanguine. Dorinţa de a utiliza informaţia obţinută de la acest senzor a rezultat din proiectarea sistemelor de reglare la Centrul de cercetări medicale din cadrul Baylor College (Norman, De Bakey şi col., 1973) şi de la Institutul de transplantologie şi organe artificiale din Moscova (Lokşin şi col., 1978).

2. A doua tendinţă este bazată pe substituirea pur mecanică a reglării manuale cu regulatoare automate. Algoritmul de reglare este ales în mod empiric, pe baza estimării acţiunilor operatorilor care au participat la experimentul realizat de un grup de experţi.

Un exemplu de proiectare în cadrul acestei tendinţe este prima versiune a aparatului avansat de reglare a CIA: „VITAMEC".

3. A treia tendinţă constă în dezvoltarea sistemelor de reglare utilizînd dinamica variaţiei variabilelor fiziologice de bază ale circulaţiei. Acesta este principiul de proiectare inclus practic în toate sistemele pentru stabilizarea unei variabile, de exemplu: tensiunea arterială şi atrială (Landis şi col., 1977; Şumakov şi col., 1978). Este bine cunoscut faptul că organismele sînt lipsite de buclă de stabilizare a tensiunii atriale. Deşi în gama de condiţii fiziologice această tensiune variază nesemnificativ, totuşi ea a sugerat unor cercetători să utilizeze o buclă de servoreglare pentru a stabiliza tensiunea atrială (Hennig şi col., 1976). Cu privire la specificul funcţionării de durată a CIA în sistemul circulator sanguin, utilizarea unor asemenea parametri ridică atît dificultăţi de ordin tehnic cît şi metodologic, făcînd utilizarea unui asemenea dispozitiv nesigură şi imprecisă.

4. A patra şi ultima tendinţă în reglarea CIA constă în investigarea mecanismelor reglării ieşirilor cordului şi proiectarea algoritmilor de reglare capabili să reproducă caracteristicile de bază pentru reglarea inimii. Această tendinţă a ridicat cele mai complicate probleme deoarece pînă în prezent nu există informaţii cantitative complete privind integrarea procesului regulator asociat cu circulaţia.

Această direcţie de cercetare implică dezvoltarea unei descrieri cantitative a buclelor de reglare ale sistemului cardiovascular, proiectarea unor modele matematice adecvate şi utilizarea acestora pentru estimarea algoritmilor de reglare a CIA.

Primele eforturi în această direcţie au fost realizate la începutul anilor '70 în comun cu alte institute de către Institutul de transplantologie şi organe artificiale şi de Institutul de automatizări din Moscova (Itkin, 1978; Uşakov şi col. 1980). Actualmente sînt disponibile unele articole

privind proiectarea algoritmilor de reglare a CIA pe bază de model în institute de cercetări din S.U.A., Japonia și R.F.G. (Kamiya și col., 1975; Fudjimasa și col., 1980; Reul și col. 1975).

3.3. *Surse de informații pentru sistemul de reglare a CIA*

Informația utilizată pentru reglarea CIA se obține cu ajutorul unor elemente sensibile prin măsurarea directă și implicită a CIA și a parametrilor de circulație (fig. 13). Primul tip include:

— elemente sensibile de tensiune arterială și venoasă;

— elemente sensibile de debit al sîngelui;

— senzori de tensiune sau saturație cu O_2 pentru sîngele arterial și venos.

Al doilea tip include:

— elemente sensibile pentru măsurarea implicită a tensiunilor de intrare și ieșire a CIA prin măsurarea tensiunii pe calea de acționare pneumatică;

— elemente sensibile pentru măsurarea implicită a fluxului de sînge în CIA, prin măsurarea parametrilor acționărilor pneumatice sau poziției membranei CIA.

Obținerea informației prin măsurări bioelectrice și biomecanice în cadrul sistemului circulator se include tot în acest caz. A se observa că informația privind activitatea mecanică sau electrică a atriilor naturale se păstrează după îndepărtarea cordului.

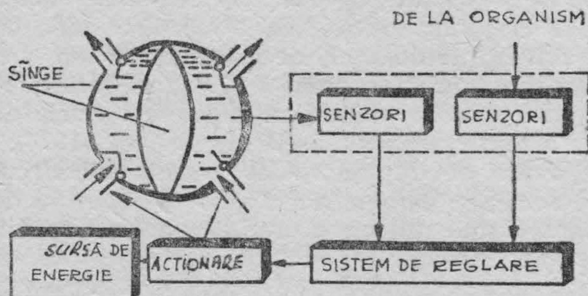


Figura 13. Surse de informații într-un sistem de reglare a unei inimi artificiale.

Realizarea practică a sistemelor de reglarea CIA cu măsurări directe este stînjenită semnificativ de dificultățile obținerii de date stabile și sigure de la senzorii implantați.

Mult mai promițătoare sînt sistemele de reglare bazate pe măsurarea implicită a parametrilor CIA prin parametri presiunii pneumatice, debitului pneumatic, sau deplasării mecanice a membranei CIA.

3.4. *Algoritmi de reglare a CIA*

Algoritmii care sînt astăzi utilizați cel mai mult sînt algoritmii utilizînd reprezentarea mecanismului Frank-Starling. Această tehnică a fost

desorisă de exemplu de către Unger (Unger și col., 1977). Sistemul automat reglează debitul de aer pe calea pneumatică, conectînd acționarea pneumatică cu camerele pneumatice ale ventriculelor. Scăderea la zero a debitului arată că avem camera sangvină ventriculară fie complet plină, fie goală. Din acest motiv, după ce tensiunea de ieșire a debitmetrului a scăzut la zero, se realizează un transfer la următorul ciclu cardiac. Astfel,

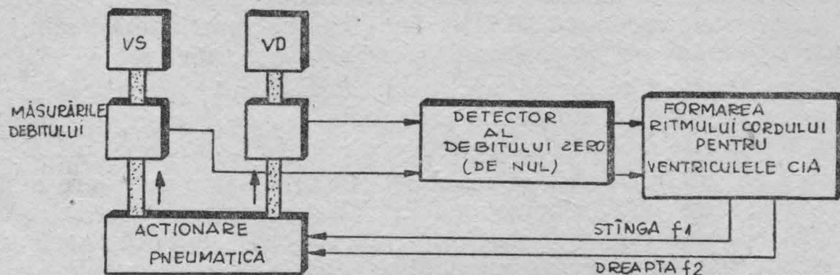


Figura 14. Schema-bloc a sistemului de reglare totală a inimii artificiale folosind mecanismul Frank-Starling: VS — ventricolul stîng, VD — ventricolul drept al CIA.

amîndouă ventriculele operează asincron cu un volum pulsatoriu maxim. O schimbare în debitul venos sau în tensiunea aortică determină fiecare ventricul să răspundă prin schimbarea pulsului cordului (fig. 14).

Încercările de a utiliza activitatea electrică a atriilor reziduale s-au concretizat în proiectarea unui sistem de reglare a CIA bazat pe microcalculator (Vajapeyam și col., 1979). Utilizînd semnalul de undă P (puls) de la electrodul implantat în reziduul atriului natural după implantarea CIA, sistemul a reglat ritmul CIA. În plus, datele privind debitul de aer și presiune din calea pneumatică a ventriculelor stîng și drept au fost introduse în calculator pentru a estima tensiunea aortică și atrială (utilizînd o tehnică similară celei a lui Rosenberg și col., 1978). Valorile măsurate ale perioadei pulsului și tensiunile aortică Pa și atrială Pb stabilesc ritmul CIA, raportul adecvat între sistolă și diastolă și presiunea aerului în căile pneumatice de reglare.

Autorii arată dificultățile în izolarea undei P pe fondul zgomotului, întrucît semnalul nu depășește 5 mV. Durata sistolei este prestabilită automat în concordanță cu raportul de mai sus pornind de la măsurarea perioadei undei P. Dacă frecvența undei P măsurate depășește 80—120 bătăi pe minut, sistemul de reglare fixează frecvența impulsului de reglare fie la nivelul cel mai ridicat, fie la nivelul cel mai scăzut.

Un sistem de reglare automată a CIA, proiectat în Centrul Medical Milton S. Hershey din cadrul Universității de stat Pennsylvania (S.U.A.) este descris de Resemberg și col. (1978). Sistemul reglează funcționarea CIA cu o ieșire în pulsație completă. Aceasta este furnizată prin fixarea duratei sistolei și presiunii pneumatice la un asemenea nivel încît com-

pleta evacuare a ventriculului artificial are loc într-o gamă largă de condiții fiziologice.

Umplerea completă a ventriculului artificial stîng este sesizată printr-un senzor interior (operînd pe baza efectului Hall), plasat în cutia rigidă a ventriculului artificial.

Ventriculul drept este comandat printr-un generator independent. Ritmul său este ales automat astfel încît tensiunea principală în ventriculul stîng este menținută la 7 mmHg. Ritmul ventriculului stîng este automat reglat invers proporțional cu tensiunea aortică principală.

Sistemul a fost testat pe un stand hidrodinamic precum și într-un experiment pe viței, demonstrînd capacitatea sa de a menține circulația sîngelui la diferite sarcini fiziologice, în particular sub exercițiu fizic.

Lucrarea lui M. Arabia și coautorii (1980) a dat o descriere a sistemului de reglare binivel-homeometric și heterometric a CIA. Reglarea heterometrică este realizată în scopul reproducerii legii lui Starling. Ea furnizează reglarea presiunii pneumatice pînă la golirea completă a ambelor ventricule. Algoritmul reglării heterometrice corelează valorile instantanee ale presiunii P(t) și volumului V(t) utilizînd binecunoscutul model muscular rheologic descris de Sonnenblick (1962). În conformitate cu algoritmul de reglare, ritmul cordului este dependent în mod liniar de volumul pulsației principale a ambelor ventricule. Pentru a schimba caracterul abrupt al curbei lui Starling, al doilea nivel va trebui inclus în cel al reglării homeometrice.

În concordanță cu algoritmul de reglare homeometrică, o creștere a tensiunii atriale principale conduce la o creștere a diferenței de presiune la intrarea valvulei ventriculului artificial.

Sistemul a fost testat pe un model fizic al sistemului circulator. Testele au demonstrat funcționarea stabilă a sistemului. Senzitivitatea la intrare a fost de un 1 l/min/mmHg, iar la ieșire de 0,04.

Cele de mai sus au condus la concluzia că actualmente sînt utilizate 3 metode principale în proiectarea algoritmilor de reglare a CIA.

Cea mai simplă metodă dintre acestea implică realizarea umplerii și golirii complete a ventriculului artificial care în oarecare măsură reproduce legea Frank-Starling. În plus, această cale este preferabilă deoarece furnizează o hemodinamică introcardiacă mai bună și o probabilitate mai scăzută în apariția trombozelor în cavitățile ventriculelor artificiale.

O altă metodă de reglare implică reglarea tensiunilor aortice și în atriul stîng. Ultimul parametru este reglat prin intermediul varierii ieșirii ventriculului drept. Astfel, fluxul în ventriculele stîng și drept sînt echilibrate.

A treia metodă de reglare a CIA, utilizează mecanismul fiziologic care asigură reglarea cardiacă. Ea implică, fie reprezentarea legilor funcționale ale mușchiului cardiac, fie utilizarea canalelor de reglare ale activității cardiace naturale (semnalul de undă).

3.5. *Estimarea eficienţei sistemului de reglare a CIA*

O tehnică de reglare poate fi preferată numai după ce toţi factorii de bază sînt recunoscuţi, iar realizabilitatea tehnică a algoritmului ales este evidentă. Aşa cum s-a arătat de către Y. Nose (Nose şi col., 1979), factorii ce realizează funcţionarea normală a acţionării pneumatice a CIA se pot clasifica în factori fiziologici şi mecanici.

Realizarea tehnică impune de asemenea serioase limitări în alegerea algoritmilor de reglare. De exemplu, nu se poate considera suficient de sigură detectarea undei P ce caracterizează activitatea electrică a atriului rezidual.

Diversitatea metodelor de proiectare a sistemelor de reglare este, în plus, limitată de lipsa unor criterii concrete pentru estimarea funcţionării lor. Marea capacitate compensatorie a organismului permite compensări temporare ale inadecvenţei circulaţiei printr-o schimbare corespunzătoare a stărilor funcţionale a celorlalte sisteme ale corpului. Astfel, de exemplu, consumarea oxigenului de către ţesuturile corpului poate fi redusă, fluxul de sînge dintre organe poate fi redistribuit. În consecinţă, sintetizarea algoritmilor de reglare ar trebui să meargă în pas cu cercetările de obţinere a metodelor optimale de estimare a acestor algoritmi şi a întregului sistem de reglare a CIA. Una dintre metodele acceptate pentru estimarea sistemului de reglarea CIA este investigarea reacţiei la următoarele perturbaţii ale unui animal avînd implantat un CIA:

— introducerea de vasopresoare şi vasodilatoare;
— varierea volumului de sînge circulator şi ocluziunea vaselor sangvine;
— varierea modului de aerisire.

În prezent, asemenea investigaţii sînt mai ales de natură calitativă. Ele studiază numai tendinţele reacţiilor fără nici o estimare cantitativă a răspunsului tranzitoriu. Un criteriu tehnic pentru estimarea sistemelor de reglare utilizînd unele criterii cantitative a fost sugerat de către Itkin în 1978. Un set de criterii a fost propus pentru estimarea performanţelor algoritmilor de reglare a CIA în termeni de senzitivitate a ieşirii cardiace la:

— consumul de oxigen al organismului;
— suprasarcina arterială;
— variaţii în volmul de sînge circulat;
— tensiune atrială (denivelarea caracteristicii de pompare a cordului).

Numeroase studii privind răspunsul sistemului cardiovascular la solicitări au arătat că cele mai mari schimbări ale ieşirii cardiace la cantităţi dozate ale consumului de oxigen sînt datorate stării de sănătate a individului. De aici s-a obţinut primul criteriu ce poate servi la estimarea indirectă a capacităţii funcţionale a CIA implantat.

O ieşire cardiacă este binecunoscută ca fiind invariantă la suprasarcină (Anrep, 1912). Studiind reacţia ieşirii cardiace la variaţia contratensiunii arteriale, acest autor a determinat că creşterea rezistenţei şi ten-

siunii arteriale conduce la creşterea forţei de contracţie. Astfel, senzitivitatea CIA la impedanţa de intrare poate fi folosită ca o măsură a asemănării cu cordul natural.

Dependenţa ieşirii cardiace de volumul de sînge circulator prezintă interes numai în faza iniţială a introducerii CIA în organism şi în timpul funcţionării în prezenţa hemoragiei.

Ultimul criteriu defineşte în esenţă denivelarea caracteristicii Starling a cordului artificial şi este utilizată practic în toate sistemele de reglare cunoscute.

3.6. *Discuţie*

Sarcina creării CIA necesitînd eforturi internaţionale ale specialiştilor din diferite domenii, reprezintă cel mai tipic exemplu de proiectare a mijloacelor tehnice pentru menţinerea vieţii. După ce se vor rezolva problemele tehnice şi chirurgicale, primele probleme cu care ne vom confrunta vor fi relative la reglare. Actualmente este evident că reglarea unui CIA ar trebui să necesite un număr minim de parametri.

Măsurarea parametrilor hemodinamicii prin tehnici directe este stînjenită atît de obstacole tehnice cît şi biomedicale. Cu atît mai vitală este atunci problema proiectării tehnicilor de măsurare indirectă (de exemplu, bazate pe măsurări pe cale pneumatică).

Deoarece sistemele de reglare a CIA, utilizate în experimentele biomedicale sînt în mod uzual externe corpului, există o tendinţă de creştere a utilizării în timp real a microcalculatoarelor. Ar trebui evidenţiat că aplicarea microcalculatoarelor implică proiectarea unui complex de cercetări, în scopul verificării şi actualizării diferiţilor algoritmi de reglare.

Selectarea algoritmilor de reglare a CIA este încă o problemă nerezolvată. Rezolvarea sa în bune condiţii necesită pe de o parte o adîncă analiză a sistemelor fiziologice ale organismului, strîns asociată cu activitatea cardiacă, iar pe de altă parte problema optimizării algoritmilor de reglare devine foarte importantă în măsura în care asigură fiabilitatea tehnică şi în măsura în care satisface necesităţile fiziologice ale organismului în fluxul sangvin.

Ar trebui notat în mod special că pentru a simplifica sarcina realizării tehnice a algoritmilor de reglare a CIA, aceştia pot diferi de particularităţile cordului natural. De exemplu, creşterea ieşirii cardiace naturale este obţinută prin creşterea ritmului cordului şi a volumului de pulsaţii. În acelaşi scop un CIA poate utiliza un singur parametru de reglare.

Concluzie

Ştiinţa modernă, în general, şi nu numai ştiinţa reglării, insistă în a descoperi fenomenele naturale: activitatea vitală a organismului şi sistemele biologice şi în a înţelege natura biologică a omului.

Autorii au dorit să arate în această lucrare diversele moduri de aplicare a tehnicilor şi mijloacelor de reglare pentru a prezenta generalizarea domeniului reglării în şi pentru organismele vii.

Știința supraviețuirii organismului este încă la începuturile ei. Este foarte clar însă că dezvoltarea în perspectivă a acestui domeniu este determinată de progresele în știința reglării.

„Toate lucrurile sînt părți ale omului; toate sînt pentru om“ scria marele scriitor rus Maxim Gorki*.

Rezultatele obținute în teoria reglării au folosit întotdeauna omului pentru a putea conduce sistemele mecanice, chimice, tehnologice. Pașii înainte ai acestei teorii constituie o bază solidă pentru a studia reglarea în și pentru sistemele biologice.

În Om și pentru Om!

Bibliografie

Anrep, G. V. (1912). On the part played by the suprarenals in the normal vascular reactions of the body. *J. Physiol.*, 45, 307—317.

Akhutin, V. M., ed. (1981). *Biotechnical Systems: Theory and Design.* Leningrad University Publ., Leningrad. 220 p. (in Russian).

Arabia, M., Franconi, C., Guerrisi, M., et al (1980). A new automatically controlled electric TAH. In *Tr. Amer Soc. Artif. Internat. Organs*, 26, 60—65.

Beneken, J. E. W., Blom, J. A., Jorritsma S. F. et al (1979). Prognosis, trend prediction and models in patient management, *J. Biomed. Eng.*, 1, 185—200.

Bernard, C. (1980). Constant of free life. E. Satinoff (Ed) *Thermoregulation*, Dowden, Hutchingson and Ross Inc., 10—19.

Bertalanffy, L. von. (1973). *General Systems Theory (Foundation, Development, Application).* George Brazillier, New York. 850 p.

Bodrov, V. A. (1983). Actual tasks of pilot staff workability. In *XIV Congress of All-Union Physiol. Soc.*, v. 2, Nauka Publ., Leningrad, 403—404 (in Russian).

Caceres, C. A. (1974). Electrocardiographic automation. *Giorn. Ital. Cardiol.*, 4, 1, 1—14.

Calow, P. (1976). *Biological Machines. A Cybernetical Approach to Life.* Edward Arnold, London, 134 p.

Carson, E. R., Finkelstein, L. (1981). Computer-based measurement in clinical medicine. In *Application of Microcomputers in Measurement*, J. Bozicevic, L. Finkelstein, D. Hoffman (Eds.), Birotehnica, Zagreb, 109—116.

Carson, E. R., Cobelli, C., Finkelstein, L. (1982). *Mathematical Modeling of Metabolic and Endocrine Systems.* J. Wiley, New York.

Cannon, W. B. (1932). *The Wisdom of the Body.* Kegan Paul, Trench, Trubner and Co, London, 236 p.

Cobelli, C., Federspil, G., Pacini, G. et al (1982). An integrated mathematical model of the dynamics of blood glucose and its hormonal control. *Math. Biosci.*, 58, 27—60.

Cooney, D. (1976). Modeling the body as compartments, sources and streams. In *Biomedical Engineering and Instrumentation. Biomedical Engineering Principles.* Marsel Dekker Inc., 157—226.

Fujimasa, I., Imachi, K., Ohmichi, H., et al (1980). An artificial heart dynamic model of circulatory regulation. In *ISAO/IFAC Symp. on Control Aspects of artif. Organs.* Accepted abstracts. Warsaw, Poland, 22—23.

Griselin, G. (1982). Industrial underwater technology. Paper No. 6 presented at *French—Soviet Symp. on New Technical Methods of Oil and Gas Exploitation on the Sea.* Comex Service, 26 p, (in Russian).

Grodins, F. (1963). *Control Theory and Biological Systems.* Columbia Univ. Press., New York — London, 120 p.

* „Adîncimile străfundurilor“, act. IV.

Guyton, A. C. (1971). The cells and its function (ch. 2). In *Textbook of Medical Physiology*. 4th ed., W.B. Sounders Publ., Phyladelphia, 1074 p.

Hennig, E., Grosse-Siestrup, C., Krautzberger, W. et al. (1976). The relationship of cardiac output and venous pressure in long surviving calves with total artificial heart. *Trans. Am. Soc. Artif. Internal Organs*. 24, 616—624.

Itkin, G. P. (1978). Methodical and technical aspects of artificial heart control (in Russian). In *Proc. Symp. Instrumentation for artificial Heart and Artificial Kidney*, ISAO, Brno, 267—272.

Kamiya, A., Togawa, T., Kobayashi, T. et at. (1975). Effects of unphysiological factors on cardiac output regulation during artificial heart pumping. *IEEE Trans. on Biomed. Eng.*, BME-22, 33, 114—119.

Kevin O'Kane, Edvard, A. (1977). Perspectives in clinical computing. ALV Comput., New York, 16, 127—182.

Khaitlin, A. I. (1980). Technical Equipment of extra-kidney cleaning. In *Technical Means of Extra-Kidney Blood Cleaning*. Medical Instrumentation Moscow, v.4, 7—10. (in Russian).

Kiraku, M., Chikayama, J. et al. (1977). Four-year experience in an AMHITS design, implementation and development. *MEDINFO-77 Proc. — 2nd World Conf. Med. Inform.*, Toronto, 755—758.

Koshcheev, V. C. (1981). *Physiology and Hygiene of Cold Individual Protection of Man*. Medicina Publ., Moscow, 288 p.

Kandis, D. L., Pierse, W. B., Rosenberg, G. et al. (1977). Long-term in vivo automatic electronic control of the artificial heart. *Trans. Am. Soc. Artif. Internal Organs*, 23, 513—525.

Leuner, H. (1977). Selbstkontrolle vegetativer Funktionen durch Biofeedback-Methoden (Ruckkopplungsverstarkung). *Therapiewoche*, 27, 31.

Lokshin, M. A., Burtin, V. A., Kremnev, V. A. (1978). Real-time convertor for information on artificial heart stroke volume. In *Transplantation of Organs in Experiments and Clunics. Artificial Organs*. Medicina Publ. Moscow, 165—167. (in Russian).

Milsum, J. H. (1966). *Biological Control Systems Analysis*. McGrow Hill Book Co., New York, 237 p.

Nalecz, M. (1981). Biocybernetics and biomedical instrumentation in Poland. *Med. Instrumentation*. 1, 54—61 (in Russian).

Norman, N. A., DeBakey, M. E., Noon, R. P. et al (1973). Monotoring and closed loop control of pneumatic blood pump. *Cardiovasc. Res. Center Bull.*, 12, 3—12.

Nose, M., Vakamudi, A. K., Kudo, T. et al. (1979). Comparative study of rigid vs flexible pneumatic artificial blood pumps. *Trans. Am. Soc. Artif. Internal Organs*, 25, 268—273.

Novoseltsev, V. N. (1978). *Control Theory and Biosystems*. Nauka Publ., Moscow, 320 p. (in Russian).

Novoseltsev, V. N. (1981). Homeostasis of the body: control aspects. In *28th Internat. Congress of Physiol. Sci.*, v.24. *Mathematical and Computational Methods in Physiology*. Pergamon Press and Akademiai Kiado, 3—12.

Novoseltsev, V. N. (1982). *Control at Organismic Level. Biophysics and Biocybernetics*. Preprint, Pushchino Sci. Center, Moscow, 10 p. (in Russian).

Novoseltsev, V. N. (1983). Modeling of engineering-physiological systems in medicine. In G.I. Marchuk and L. N. Belykh (Eds.) *Mathematical modeling in immunology and medicine*, North-Holland Publ. Comp. Amsterdam, 351—357.

Novoseltsev, V. N. (1983a). Engineering Physiology — new branch of control theory application. *Izmerenija, Kontrol, Avtomatizacija*, 4, 55—63. (in Russian).

Petrovsky, A. M. Novoseltsev, V. N., Sakharov, M. P. (1980). Control of artificial organs and mathematical modelling of physiological systems. *Symp. on Control Aspects of Artificial Organs*. Accepted abstracts. Warsaw, Poland, 50—52.

Pipberger, H. V. (1975). Clinical aspects of a second generation electrocardiographic computer program. *Am. J. Cardiol.*, 35, 5, 597—607.

Prosser, C. L. (Ed.), (1973). *Comparative animal physiology*. Saunders Publ., Philadelphia, 966 p.

Rashevsky, N. .1965). Models and mathematical principles in biology. In *Theoretical and Mathematical Biology*. Blaisdell Publ., New York.

Ray, C. D. (Ed.), (1974). *Medical Engineering*. Year Book Medical Publ., Chicago, 63—71.

Reul, H., Minamitani, H., Runge, J. (1975). A hydraulic analog of the systemic and pulmonary circulation for testing artificial heart. *Proc. Eur. Soc. Organs (ESAO)*, 2, 120—127.

Rosen, R. (1967). *Optimality Principles in Biology*. Butterworths, London.

Rosenberg, G., Landis, D. L., Phillips, W. M. et al. (1978). Determining arterial pressure, left atrial pressure and cardiac output from the left pneumatic drive line of the total artificial heart. *Trans. Am. Soc. Artif. Internal Organs*, 24, 371.

Savchenko, N. E., Varenikov, V. M., Shilay, P. M. (1980). Clinical effectiveness of automatic „artificial kidney". In *Technical Means of Extra-Kidney Blood Cleaning: Medical Instrumentation, News*. Inst. of Medical Instrumentation Moscow, v.4, 47—49.

Sechenov, I. M. (1952). *Reflexes of Brain. Selected Works*. v.1., USSR Academy of Sci. Publ., Moscow, (in Russian).

Shumakov, V. I., (1975). *Artificial Heart*. Znanije, Moscow, 63 p. (in Russian).

Shumakov, V. I. (1976). Artificial organs. *Medical Instrumentation*, 4, 3—5. (in Russian).

Shumakov, V. I., Khanin, M. A., Guskov, I. A. et al. (1978). Control system for artificial heart. *Medical Instrumentation*, 4, 11—14 (in Russian).

Shumakov, V. I., Tolpekin, V. E. (1980). *Assistory Circulation*, Medicina, Moscow, 248 p. (in Russian).

Sonnenblick, E. (1962). Force-velocity relation in mammalian heart muscle. *Am. J. Physiol.*, 202, 931.

Trapeznikov, V. A. (1981). Control problems in medicine. In *Fundamental Sciences — to Medicine*. Nauka, Moscow, 100—105 (in Russian).

Umansky, S. P. (1982). *Cosmonaut's Equipment*. Mashinostrojenije, Moscow, 124 p. (in Russian).

Unger, F., Deutsch, M., Eckersberger, F. et al. (1977). Artificial heart driven by an automatic driving system. *Med. Instrum.*, 11 (4), 208—211.

Ushakov, V. B., Gerasimenko, A. I., Voinov, V.E. (1978). Electronic hybrid simulating complex for investigation of artificial heart circulation (in Russian). In *Proc. Symp. Instrumentation for Artificial Heart and Artificial Kidney*, ISAO, Brno, 267—272.

Vajapeyam, B., McInnis, B. C., Everette, R. L. et al. (1979). A microcomputer based control system for artificial heart. *Trans. Am. Soc. Artif. Internal Organs*, 25, 379—382.

Viktorov, V. A. (1979). On completed technical equipment of polyclinical organisations. Paper presented at *Internat. Congress of Biomed. Instrumentation*, Berlin.

Viktorov, V. A. (1981. Modern state and main tendencies in development of medical instrumentation. *Medical Instrumentation*, 2 (in Russian).

Viktorov, V. A., Dubrovsky, E. N. (1982). Usage of systemic methodology for NIOKR planning problems and production of medical instrumentation. Paper presented at Internat. meeting *Health Care Systems Control*, Moscow.

Viktorov, C. A., Kerbunov, V. V. (1979). Some tendencies in development of medical instrumentation. *Medical Instrumentation* 4 (in Russian).

Vishniakov, V. A., Merenov, I. V. (1982). *Deep Underwater Diving Equipment*. Sudostrojenije, Leningrad, 240 p. (in Rusnian).

Zorile, V. I. (1983). Physiological aspects of optimization in operator training. In *XIV Congress of All-Union Physiol. Soc.*, v.2, Nauka Publ., Leningrad, 407 (in Russian).

CS6. O NOUĂ FUNDAMENTARE TEORETICĂ ȘI ALGORITMICĂ PENTRU ESTIMARE, IDENTIFICARE ȘI INFORMARE

P. Kovanic

Institutul de Teoria Informației și Automatică, Academia Cehoslovacă de Științe, Praga, Cehoslovacia

Rezumat. Se expune succint o nouă teorie a sistemelor cognitive cantitative. Această așa-numită *teorie gnostică* deduce din două axiome simple un model matematic al culegerii datelor perturbate de o incertitudine al cărei model statistic este necunoscut sau chiar nejustificabil. Teoria gnostică a eșantioanelor mici de date sărace (în sens informațional — N.T.) se bazează pe legi guvernînd incertitudinea fiecărei date individuale cum ar fi principiile variaționale ale cinematicii virtuale a datelor reale și a dinamicii lor strîns legate atît de entropia cît și de informația datelor. Existența și cunoașterea unui ciclu gnostic ideal pentru fiecare dată reală permite să se elaboreze algoritmi pentru prelucrarea optimală a datelor. Programele de calcul bazate pe algoritmii gnostici maximizează informația obținută din date și furnizează caracteristici ale datelor a căror robustețe sau sensibilitate în raport cu datele anormale sau normale este optimală. Cu cît sînt datele mai proaste, cu atît sînt mai utile aceste caracteristici față de cele statistice clasice. Domeniul de aplicare include estimarea ambilor parametri de poziție și de scală ai eșantioanelor de date și a corelațiilor lor generalizate, estimarea probabilității și o estimare neparametrică a distribuției de probabilitate, filtrarea discretă neliniară, predicție și netezire, identificarea sistemelor cu perturbații puternice și poziționarea adaptivă a sistemelor de alarmă.

Cuvinte cheie. Sisteme cognitive; culegere de date; prelucrarea datelor; regăsirea informației; modelare; filtrare neliniară; estimarea parametrilor; identificare; sisteme cu eșantionare; teorie gnostică; cibernetică relativistă.

Introducere și notații principale

Conducerea este o interacțiune deliberată între subiect și obiect, care include în mod necesar estimarea mărimilor reale și identificarea obiectului. Cu cît cunoașterea este mai bună, cu atît este mai bună conducerea. Dar procesele cognitive reale sînt în mod inevitabil perturbate de incertitudini atît de origine cît și de natură diferite. Situațiile practice nu sînt de obicei nici staționare, nici ergodice și rareori este posibilă o descriere statistică completă a incertitudinilor, datorită lipsei de timp, lipsei de date și inadecvanței unei abordări statistice.

De aceea, a fost elaborată o teorie mai generală a incertitudinii, *teoria gnostică* făcînd posibilă tratarea unor eșantioane mici de date reale sub influența puternică a unei incertitudini care poate să nu aibă un model statistic. Teoria deduce formulele informației purtate de fiecare dată individuală pornind de la o axiomă foarte rezonabilă. Ea ajunge la un ciclu cognitiv închis („ciclul gnostic ideal") care este o analogie cognitivă a bine-cunoscutului ciclu Carnot al termodinamicii. Teoria demonstrează o lege de echivalență descriind cantitativ interdependența mutuală între informația unei date și entropia termodinamică. Ea demonstrează, de asemenea, că ambele legi ale termodinamicii se mențin pentru ciclul gnostic ideal și că acest ciclu este optimal în sensul că minimizează atît creșterea inevitabilă a entropiei termodinamice cît și pierderea inevitabilă de informație în cadrul ciclului. Nu sînt necesare nici un fel de ipoteze asupra distribuțiilor statistice ale incertitudinii pentru a evalua numeric informația unui eșantion de date. Dacă există un motiv de a identifica distribuția incertitudinii reprezentată de un eșantion de date, atunci din teoria gnostică pot fi deduși, de asemenea, algoritmi pentru o estimare directă din date a unor astfel de distribuții. Ideea este: „Lăsați datele să vorbească pentru ele însele!" Importanța practică a teoriei gnostice poate fi înțeleasă în faptul că generează formule de estimare și algoritmi care sînt fie robuști fie sensibili în raport cu datele anormale sau normale (într-un grad optimal). Sînt disponibile de asemenea în cadrul

teoriei gnostice versiuni robuste/sensibile ale parametrilor de poziţie şi de scală a eşantioanelor de date, a corelaţiilor lor, a probabilităţii şi a ponderilor datelor.

O expunere completă a teoriei gnostice este prezentată în [1]—[3]. Scopul acestui articol este de a rezuma ideile şi rezultatele principale ale acestei teorii în măsura care ar putea atrage atenţia utilizatorilor potenţiali ai algoritmilor rezultaţi din teorie.

Se impun aici cîteva menţiuni. După cum se arată mai jos, teoria gnostică conduce la formule cu o ponderare individuală a datelor. Datelor individuale li se atribuie ponderi în funcţie de incertitudinea fiecărei date particulare. Se va arăta că această caracteristică este un analog al dependenţei de viteză a masei unei particule relativiste. Există o altă analogie între tensorul energie-impuls ataşat unei astfel de particule şi tensorul gnostic caracterizînd incertitudinea datelor. O altă analogie este între timpul propriu sau durata unui eveniment relativist şi „timpul" gnostic caracterizînd efectul incertitudinii asupra unei date. Toate aceste analogii susţin afirmaţia că există o legătură profundă între teoria gnostică a datelor incerte şi fizica relativistă. Evenimentele incerte considerate de teoria gnostică au o natură generală. Totuşi, o teorie generală a incertitudinii trebuie să se aplice de asemenea într-un caz particular simplu al incertitudinii legate de mişcarea fizică. Corespondenţa unor astfel de teorii aparent diferite nu este deci surprinzătoare. Ea este o înfăptuire a principiului de corespondenţă al lui Bohr. Ideea relaţiilor între fizica relativistă şi cibernetică a apărut probabil pentru prima oară într-un articol al lui G. Jumarie care a publicat o serie de articole asupra ciberneticii relativiste (vezi [4] şi [5]). Ideea lui Jumarie poate fi probabil formulată ca o necesitate de a lua în considerare subiectivitatea observatorului în toate consideraţiile cibernetice. G. Jumarie aduce argumente teoretice de o natură foarte generală pentru a-şi susţine punctul său de vedere. Ele diferă de cele ale teoriei gnostice. Cu toate acestea, rezultatele menţionate ale teoriei gnostice sprijină de asemenea ideea ciberneticii relativiste.

Ideea asupra dependenţei informaţiei de obiectivitatea receptorului este chiar mai veche. Această problemă a fost formulată şi rezolvată de A. Perez [6] care a introdus canalul receptorului informaţiei.

Ponderile individuale ale fiecăreia din date apar de asemenea în cadrul teoriei statistice robuste, desigur fără referiri la unele concepte relativiste.

Cinematica gnostică a unei date individuale

Un sistem „obiect-subiect" angajat într-un proces de cunoaştere cantitativă este *„sistemul gnostic cantitativ"*. El include două faze: *cuantificarea* şi *estimarea*. Cuantificarea este o transformare de la mărimile reale la numere reale numite *date reale*. Există numai două proceduri de cuantificare, *măsurarea* mărimilor reale şi *numărarea* obiectelor reale avînd aceeaşi proprietate. Rezultatul unei măsurări spune de cîte ori o mărime măsurată depăşeşte o unitate de măsură sau invers. Deci, rezultatele măsurării sînt pozitive şi finite, la fel ca şi un număr de obiecte numărate. În consecinţă, toate rezultatele cuantificării sînt numere reale finite pozitive, ele sînt (dintr-un punct de vedere) „imagini" inexacte dar cunoscute ale unei mărimi „originale" reale dar necunoscute. *Estimarea* este o transformare a datelor reale în estimaţii ale mărimii cuantificate. Un subiect poate îmbunătăţi calitatea proceselor de cuantificare numai în întregime şi a priori cu ajutorul unei tehnici mai bune de măsurare şi comunicare, prin perfecţionările experimentului ş.a.m.d. Dar un proces de cuantificare particular şi incertitudinea care îl perturbă nu sînt sub controlul subiectului. Procesul de estimare este o chestiune de strategie a subiectului, „mutarea" lui într-un joc contra naturii. Cu cît este mai bună cunoaşterea regulilor acestui joc, cu atît mai bună este interpretarea datelor. Să trecem deci la regulile „jocului gnostic".

Noţiunea de bază este cea a *eşantionului de date reale*. Aceasta este un n-tuplu

$$Z \equiv Z(z_0, n) := \langle z_1, \ldots, z_n \rangle \quad (1 < n \leqslant \infty) \qquad (1)$$

unde z_i sînt rezultatele cuantificării practice a unei mărimi Q, rezultatul cuantificării căreia în condiții ideale, fără incertitudine, ar fi z_0 (numită *valoarea ideală*).

Să notăm R_+ și R_1 următoarele intervale deschise de numere reale

$$R_+ := (0, \infty) \qquad R_1 := (-\infty, \infty) \tag{2}$$

Axioma 1 a teoriei gnostice afirmă: Fie z sau valoarea ideală z_0 sau o dată reală z_i $(i=1, \ldots n)$ dintr-un eșantion de date $Z(z_0, n)$ sau un rezultat posibil arbitrar al cuantificării. Atunci

$$z = z_0 \zeta \qquad (z_0 \in R_+, \ \zeta \in R_+). \tag{3}$$

În raport cu această exiomă este convenabilă următoarea parametrizare a efectului incertitudinii

$$z := z_0 \, e \, \Omega \qquad (\Omega \Cup R_1) \tag{4}$$

Aceasta este o reprezentare „polară" pe un plan care are o geometrie încă necunoscută. Este util să trecem la un alt sistem de coordonate

$$x := z_0 \mathrm{ch}\Omega \qquad y := z_0 \mathrm{sh}\Omega \tag{5}$$

și să facem uz de notația matricială

$$u := \begin{bmatrix} x \\ y \end{bmatrix} \quad {}_c u := \begin{bmatrix} y \\ x \end{bmatrix} \quad {}_c^c u := \begin{bmatrix} -y \\ x \end{bmatrix} \quad u_0 := \begin{bmatrix} z_0 \\ 0 \end{bmatrix} \quad {}_c u_0 := \begin{bmatrix} 0 \\ z_0 \end{bmatrix} \tag{6}$$

$$K_q(\Omega) \equiv K_q := \begin{bmatrix} \mathrm{ch}\,\Omega & \mathrm{sh}\,\Omega \\ \mathrm{sh}\,\Omega & \mathrm{ch}\,\Omega \end{bmatrix} \tag{7}$$

Rezultă imediat

$$u = K_q(\Omega) u_0 \qquad {}_c u = K_q(\Omega) {}_c u_0. \tag{8}$$

Matricea K_q poate fi numită *canalul de cuantificare*. El transformă vectorul ideal u_0 sau ${}_c u_0$ din (8) în u și ${}_c u$. Toți acești vectori îi numim *evenimente gnostice*. Dacă u' și u'' sînt două evenimente gnostice aparținînd aceluiași eveniment gnostic ideal u_0, atunci se poate arăta că au loc transformările

$$u'' = K_q(\Omega) u' \qquad {}_c u'' = K_q(\Omega) {}_c u' \tag{9}$$

unde

$$\Omega = \Omega'' - \Omega' \tag{10}$$

pentru parametrii canalelor. Canalele de cuantificare transformă astfel în caz general un eveniment gnostic arbitrar într-altul. Parametrul Ω al canalului caracterizează relația între cele două evenimente datorată efectului incertitudinii. Produsul matricial a două canale are parametrul egal cu suma parametrilor ambilor factori. În consecință, mulțimea G_q a tuturor canalelor de cuantificare posibile este un grup comutativ în raport cu multiplicarea canalelor care corespunde unei cascade a acestor canale. Grupul G_q este o transformare izomorfă a grupului aditiv al parametrilor Ω. Acest parametru este o variabilă independentă a procesului de quantificare. El joacă rolul unui tip special de „timp", de timp de cuantificare, de măsură a „distanței" între două evenimente gnostice. Atunci grupul G_q este un model matematic al „mișcării" gnostice provocate de incertitudine. Efectul incertitudinii este bipolar, de aceea variabila „timp" Ω poate descrește sau la fel de bine crește. Aceasta este o însușire bună dînd posibilitatea de compensare mutuală a diferite mișcări gnostice. O caracteristică interesantă a tuturor canalelor de cuantificare poate fi văzută din (7):

$$\mathrm{Det}\,\{K_q\} = 1. \tag{11}$$

Să combinăm evenimentele de ambele tipuri (*primul* tip u și *al doilea* tip $_cu$) într-o matrice $U=(u, {}_cu)$. Atunci (9) este

$$U''=K_q(\Omega)U' \tag{12}$$

și

$$\text{Det}\,\{U''\}=\text{Det}\,\{U'\}=z_0^2 \tag{13}$$

datorită lui (11). Aceasta arată că valoarea adevărată, ideală z_0 este un invariant al grupului mișcărilor de cuantificare. Aceasta ne conduce la problema unei metrici. Nu a fost presupus nimic explicit asupra metricii pe varietatea evenimentelor u și $_cu$. Utilizînd noțiunea lui Riemann de spațiu metric (o metrică definită de un tensor metric pe o varietate) ne întrebăm dacă există o libertate în alegerea unui tensor metric pe varietatea evenimentelor gnostice cînd metrica (un produs scalar) rămîne invariantă sub grupul G_q. Răspunsul s-a dovedit a fi negativ. Că există exact o metrică de acest tip, una minkowskiană, este deja furnizat de consecințele Axiomei 1. Produsul scalar a două evenimente de primul tip are în această metrică forma

$$s_{12}=u_1^T g_M u_2 \tag{14}$$

unde

$$=g_M\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix} \tag{15}$$

este reprezentarea matricială a tensorului metric minkowskian. Produsul scalar a două evenimente de al doilea tip diferă clar de (14) numai prin semn. Mărimea $\text{Det}\,\{U\}=z_0^2$ este astfel patratul duratei evenimentului u. Ea rămîne invariantă deoarece canalele K_q reprezintă rotația ortogonală (minkowskiană) a evenimentelor u (sau $_cu$) fără o modificare a duratei lor (minkowskiene) egală cu

$$z_0=\sqrt{x^2-y^2} \tag{16}$$

în cazul evenimentelor de primul tip și $\sqrt{-1}\ z_0$ în cazul evenimentelor de al doilea tip.

Întrucît tensorul metric al spațiului este cunoscut, este posibil să se studieze liniile geodezice. Se poate demonstra că există două familii de linii geodezice în cazul particular în considerare: linii drepte și cercuri minkowskiene. Înseamnă că „mișcarea" de cuantificare gnostică de-a lungul unei traiectorii corespunzînd unor canale de cuantificare trebuie să satisfacă un principiu variațional. Într-adevăr, se poate arăta că modulul $|\Omega|$ al „timpului" de cuantificare Ω este o durată relativă a unei astfel de traiectorii și că este durata relativă maximă posibilă a acestei traiectorii. Incertitudinea maximizează astfel o distanță de la datele reale la valoarea ideală.

Un subiect învățînd din cinetica cuantificării să-și aleagă strategia pentru estimare se interesează dacă există un canal dual celui de cuantificare. Un astfel de canal trebuie să fie atît simetric ca și canalul de cuantificare satisfăcînd cele două ecuații (9) cît și necondiționat regulat (11). Există tocmai un canal avînd aceste caracteristici și operînd în planul complex. Întorcîndu-ne la varietatea reală a evenimentelor gnostice cu ajutorul unei transformări liniare particulare obținem o versiune reală a *canalului de estimare*

$$K_e(\omega)=\begin{bmatrix}\cos\omega & -\sin\omega\\ \sin\omega & \cos\omega\end{bmatrix} \tag{17}$$

parametrizat de o variabilă reală ω. Transformarea de estimare este astfel

$$u''=K_e(\omega)u. \tag{18}$$

Mulțimea tuturor matricilor posibile K_e este de asemenea un grup comutativ în raport cu înmulțirea matricilor. El reprezintă un grup de „mișcări" de estimare,

„timpul" de estimare fiind parametrul ω. Invariantul metric unic sub grupul G_e este unul euclidian. Invariantul grupului este mărimea

$$r := \sqrt{x^2+y^2}, \tag{19}$$

raza (euclidiană) a unui cerc (euclidian). Mișcarea de estimare este astfel rotația ortogonală (euclidiană) a evenimentelor gnostice. Durata relativă a traiectoriei de-a lungul cercului este $|\omega|$. Ea este minimul duratelor variatelor traiectorii. O astfel de alegere a canalului de estimare face ca estimarea să fie de fapt un proces dual celui de cuantificare.

Pentru a închide un sistem al acestor transformări poate fi introdus un al treilea tip de canale gnostice: canalul de atenuare/amplificare care are forma unei matrici

$$K := \begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix} \quad (k \in R_+). \tag{20}$$

Mulțimea G_a a tuturor canalelor posibile K_a este, de asemenea, un grup comutativ. Se poate vedea că acest grup reprezintă o mișcare reală generalizată deoarece este legat de variațiile mărimii ideale z_0. Ambele mișcări gnostice sînt ortogonale mișcării reale generalizate în raport cu „propria" lor metrică. Cele trei tipuri de mișcări sînt rezumate în Tabelul 1:

Tabelul 1 **Trei grupuri gnostice**

Grup		Timp	Canal	Invariant	Variabile dependente
Simbol	Mișcare				
G_2	Cuantificare	Ω	K_o	$\sqrt{x^2-y^2}$	$\sqrt{x^2+y^2}$, $\frac{y}{x}$
G_e	Estimare	ω	K_e	$\sqrt{x^2+y^2}$	$\sqrt{x^2-y^2}$, $\frac{y}{x}$
G_a	Atenuare/Amplificare	z_0 r	K_a	$\frac{y}{x}$	$\sqrt{x^2+y^2}$, $\sqrt{x^2-y^2}$

Avem într-adevăr trei mișcări mutuale complementare. Ele pot fi utilizate pentru a crea următorul ciclu gnostic închis: *Ciclul gnostic ideal* $(CGI)_i$ definit de o dată reală z_i $(z_i \neq z_0)$ este un triplet de segmente continue de linii interconectînd succesiv trei puncte ale varietății evenimentelor gnostice, și anume

— un segment al cercului minkowskian de la $(z_0, 0)$ la $(z_0 \text{ cu } \Omega_i, z_0 \text{ sh } \Omega_i)$

— un segment al cercului euclidian de la $(z_0 \text{ ch } \Omega_i, z_0 \text{ sh } \Omega_i) \equiv (r_i \cos \omega_i, -r_i \sin \omega_i)$ la $(r_i, 0)$

— un segment al liniei drepte de la $(r_i, 0)$ la $(z_0, 0)$.

Există un complementar al acestui ciclu $(CGI)_i$ trecînd prin punctele corespunzătoare lui ${}_cu_i$ în locul lui u_i. În ambele cazuri

$$r_i^2 = z_0^2 \text{ch} 2\Omega_i = x_i^2 + y_i^2 \text{ și}$$

$$z_0 \text{ch}\Omega_i = r_i \cos \omega_i \qquad z_0 \text{sh}\Omega_i = -r_i \sin \omega_i \tag{21}$$

„Idealitatea“ lui CGI are două aspecte:

1) În practică nu cunoaștem niciodată exact pe z_0, deci nu sîntem niciodată capabili să închidem de fapt ciclul.

2) CGI este cel mai bun dintre ciclurile gnostice posibile trecînd prin aceleași puncte „muchie“. Pentru a arăta aceasta este necesar să trecem la dinamică.

Dinamica gnostică a unei date reale individuale

O importanță fundamentală în studiile dinamice o au patratele canalelor

$$K_q^2(\Omega) \equiv K_q(2\Omega) \qquad K_e^2(\omega) \equiv K_e(2\omega) \tag{22}$$

Se poate arăta că aceste mărimi sînt transformate ca tensori cînd evenimentele gnostice sînt transformate de canalele gnostice. Pentru a înțelege mai bine rolul acestor tensori introducem întîi conceptul de neasemănare a evenimentelor.

Se spune că două evenimente u' și u'' sînt σ, ε-asemenea dacă

$$\frac{x'}{y'} = \sigma \left(\frac{x''}{y''}\right)^{\varepsilon} \quad \text{unde } \sigma = \pm 1, \quad \varepsilon = \pm 1. \tag{23}$$

Dacă un cuplu de evenimente nu satisface (23) atunci evenimentele sînt neasemenea într-un grad care este măsurabil prin diferența ambilor membrii ai lui (23) sau printr-o mărime proporțională cu această diferență. Utilizînd (23) și (21) obținem astfel măsuri normalizate de neasemănare a evenimentelor rezumate în Tabelul 2:

Tabelul 2 **Măsuri de neasemănare a evenimentelor u′ și u″**

Asemănare		Măsura neasemănării	
σ	ε	Minkowskiană	Euclidiană
+1	+1	$\mathrm{sh}\,(\Omega''-\Omega')$	$\sin\,(\omega''-\omega')$
+1	−1	$\mathrm{ch}\,(\Omega''-\Omega')$	$\cos\,(\omega''+\omega')$
−1	+1	$\mathrm{sh}\,(\Omega''+\Omega')$	$\sin\,(\omega''+\omega')$
−1	−1	$\mathrm{ch}\,(\Omega''+\Omega')$	$\cos\,(\omega''-\omega')$

Astfel produsele canalelor $K_q(\Omega'')K_q(\Omega') = K_q(\Omega''+\Omega')$ și $K_e(\omega'')K_e(\omega') = K_e(\omega''+\omega')$, care sînt membre ale grupurilor G_q și G_e așa cum sînt și canalele, caracterizează neasemănarea evenimentelor corespunzătoare acestor canale. Cazul lor special K_q^2 și K_e^2 obținut pentru $\Omega' = \Omega''$ (sau $\omega' = \omega''$) caracterizează astfel neasemănarea unui eveniment cu el însuși, „prin valoarea ideală“. Nu este nimic ciudat în această noțiune: o persoană stînd în fața unei oglinzi poate măsura distanța între ea însăși și propria ei imagine arătată de oglindă ca distanța între ea (și oglindă — N.T.) multiplicată cu 2. O astfel de măsură va caracteriza relația între persoană și oglindă. Tensori K_q^2 și K_e^2 (tensorii „gnostici“) caracterizează astfel relațiile între un eveniment și valoarea ideală. Ei pot fi scriși sub forma matricilor

$$K_q^2 = \begin{bmatrix} \frac{1}{f} & h_q \\ h_q & \frac{1}{f} \end{bmatrix} \qquad K_e^2 = \begin{bmatrix} f & -h_e \\ h_e & f \end{bmatrix} \tag{24}$$

utilizînd următoarele noţiuni:

$$\textit{Fidelitate f: } = \cos 2\omega = \frac{x^2-y^2}{x^2+y^2} \tag{25}$$

$$\textit{Irelevanţa de estimare } h_e := \sin 2\omega = -\frac{2xy}{x^2+y^2} = -\mathrm{th}\, 2\Omega \tag{26}$$

$$\textit{Irelevanţa de cuantificare } h_q := \mathrm{sh}\; 2\omega = \frac{2xy}{x^2-y^2} = -\mathrm{tg}\, 2\omega \tag{27}$$

$$\textit{Infidelitate } \frac{1}{f} = \mathrm{ch}\, 2\Omega \tag{28}$$

Toate relaţiile dintre mărimi decurg din (21). Aceste componente ale tensorilor gnostici reprezintă cheia studiilor dinamice. Utilizînd operatorul lui Laplace ∇^2 (care are în metrica minkowskiană forma $\nabla_M^2(.) = \frac{\partial^2(.)}{\partial x^2} - \frac{\partial^2(.)}{\partial y^2}$ şi în metrica euclidiană $\nabla_E^2(.) = \frac{\partial^2(.)}{\partial x^2} + \frac{\partial^2(.)}{\partial y^2}$) ajungem la ecuaţia undelor

$$\nabla_M^2\left(\frac{1}{f}\right) + \frac{4}{x^2-y^2}\,\frac{1}{f} = 0, \quad \nabla_E^2(f) + \frac{4}{x^2+y^2}\, f = 0 \tag{29}$$

demonstrînd că atît infidelitatea cît şi fidelitatea (interpretate drept cîmpuri scalare peste varietatea evenimentelor gnostice) difuzează de la un punct la altul. Se poate arăta că lungimea vectorului fluxului de difuzie a acestor mărimi este proporţională cu irelevanţele respective. Tensorii gnostici pot fi astfel interpretaţi ca tensori de „infidelitate-flux de infidelitate" şi de „fidelitate-flux de fidelitate". Ambele ecuaţii (29) sînt valabile pentru toate punctele varietăţii. Dar să considerăm numai cercurile corespunzătoare traiectoriilor evenimentelor interioare canalelor pentru un z_0 fixat sau un r fixat. Punctele acestor traiectorii vor fi notate x' şi y'. Primii termeni din (29) reprezintă atunci distribuţia ambelor cîmpuri 1/f şi f de-a lungul traiectoriilor corespunzătoare canalelor. Termenii secunzi pot fi în consecinţă rescrişi pentru a obţine forma surselor altor cîmpuri scalare I_q şi I_e peste intervalele de irelevanţe

$$-\infty \leq h_q < \infty \qquad -1 < h_e < 1.$$

Echivalenţele surselor ambelor tipuri de cîmpuri scalare iau atunci forma

$$\left[\nabla_M^2\left(\frac{1}{f}\right)\right]_{x',\,y'} = c_1 \left[\frac{d^2 I_e}{dh_e^2}\right]_{x',\,y'} \quad (x'^2 - y'^2 \text{ şi } c_1 \text{ sînt constante})$$

$$\left[\nabla_E^2(f)\right]_{x',\,y'} = c_2 \left[\frac{d^2 I_q}{dh_q^2}\right]_{x',\,y'} \quad (x'^2 + y'^2 \text{ şi } c_2 \text{ sînt constante}) \tag{30}$$

Mărimile I_q şi I_e numite *variaţia informaţiei prin cuantificare* şi respectiv *prin estimare,* sînt astfel definite explicit de membrii drepţi din (31)* ca

$$I_q := -\int_0^{h_q}\int_0^{\eta} \frac{(d\eta)^2}{1+\eta^2} \qquad I_e := \int_0^{h_e}\int_0^{\eta} \frac{(d\eta)^2}{1-\eta^2} \tag{31}$$

sau — după integrare —

$$I_q = H(1/2) - H(p_q) \qquad I_e = H(1/2) - H(p_e) \tag{32}$$

---

* În original este referită, în mod eronat, relaţia (3.9) — N.T.

unde

$$H(p) = -p \ln p - (1-p)\ln(1-p) \quad (33)$$

și

$$p_q = {}^1/_2(1 - \sqrt{-1}\, h_q) \qquad p_e = {}^1/_2(1 - h_e). \quad (34)$$

O analiză arată că mărimile I_q și I_e au într-adevăr proprietățile modificării prin cuantificare și estimare a informației asupra valorii ideale z_0 purtată de o dată egală cu z. Parametrii p_q și p_e determină măsuri generalizate ale intervalelor întocmai cum determină și probabilitatea $P(z < z_0)$.

Traiectoriile circulare corespunzătoare lui CGI sînt spații riemanniene unidimensionale a căror metrică este deja determinată. Aceste traiectorii satisfac condiția variațională importantă

$$\oint_{CGV_i} dI \leqslant \oint_{CGI_i} dI < 0 \quad (35)$$

unde CGV_i este un ciclu gnostic închis diferit trecînd prin aceleași puncte „muchie" determinate de o dată z_i și de z_0 ($z_i \neq z_0$) ca și ciclul ideal CGI_i. Variabila dI este dI_q în timpul cuantificării, dI_e în timpul estimării și zero în timpul atenuării. Ciclul ideal este astfel cea mai bună traiectorie. El minimizează pierderea de informație datorită unui ciclu gnostic închis. Negativitatea variației globale a informației într-un ciclu are o semnificație analogă celei a legii a doua a termodinamicii dar ea este asociată cu un proces cognitiv.

Există o strînsă legătură între procesele gnostice și termodinamică. Se poate arăta [3] că între variațiile ΔS_q și ΔS_e ale entropiei termodinamice în cursul cuantificării și estimării și infidelitate respectiv fidelitate au loc următoarele relații

$$\Delta S_q = c\left(\frac{1}{f} - 1\right) \qquad \Delta S_e = c(f-1) \quad (36)$$

(unde c este constantă) și că pentru entropia termodinamică S are loc un analog al lui (35)

$$\oint_{CGV_i} dS \geqslant \oint_{CGI_i} dS > 0. \quad (37)$$

Egalitățile (36) oferă a interpretare suplimentară a tensorilor gnostici ca tensori entropie-flux de entropie. Substituirea lui (36) în (30) furnizează o descriere cantitativă a interdependenței între entropia termodinamică și informație.

O observație importantă: toate mărimile menționate, informația unei singure date z_i, măsura de tip probabilitate ca și entropia sînt ușor calculabile numeric ca funcții de raportul z_i/z_0.

Legea de compoziție a datelor

O problemă crucială a fiecărei teorii a incertitudinii este legea de compoziție. *Axioma 2* a teoriei gnostice afirmă următoarele:

$$K_q(2\Omega_c) = \frac{1}{w_q}\sum_1^n K_e(2\Omega_i) \qquad K_e(2\omega_c) = \frac{1}{w_e}\sum_1^n K_c(2\omega_i) \quad (38)$$

unde $K_q(2\Omega_i)$ și $K_e(2\omega_i)$ sînt tensorii gnostici corespunzători datelor z_i dintr-un eșantion de date $Z(z_0, n)$, Ω_c și ω_c sînt parametrii unui eveniment compus iar w_q și w_e

sînt ponderile normalizate ale tensorilor compuși $K_q(2\Omega_c)$ și $K_e(2\omega_c)$. Din condiția de normalizare (11)

$$\mathrm{Det}\{K_q(2\Omega_c)\}=\mathrm{Det}\{K_e(2\omega_c)\}=1 \tag{39}$$

rezultă că

$$w_q=\sqrt{\left(\sum_i^n \mathrm{ch}\,2\omega_i\right)^2-\left(\sum_i^n \mathrm{sh}\,2\Omega_i\right)^2}$$

$$w_e=\sqrt{\left(\sum_i^n \cos\,2\omega_i\right)^2+\left(\sum_i^n \sin\,2_i\omega_i\right)^2} \tag{40}$$

Parametrii Ω_i și ω_i ai evenimentelor primare determinate direct din datele z_i sînt supuși condiției (21) rescrisă ca th $\Omega_i=-$tg ω_i. O astfel de relație simplă nu se menține pentru parametrii Ω_c și ω_c ai evenimentului compus. Ei sînt determinați din (38) de evenimentele primare

$$\mathrm{th}\,2\Omega_c=\left(\sum_i^n \mathrm{sh}\,2\Omega_i\right)\left(\sum_i^n \mathrm{ch}\,\Omega_i\right)^{-1}$$

$$\mathrm{tg}\,2\omega_c=\left(\sum_i^n \sin\,2\omega_i\right)\left(\sum_i^n \cos\,2\omega_i\right)^{-1} \tag{41}$$

Merită a fi menționate patru aspecte importante ale legii de compoziție (38) deoarece ele susțin alegerea Axiomei 2:

1) Există o corespondență strînsă între evenimentele gnostice și evenimentele mecanicii relativiste în cazul particular în care obiectele cuantificate au o natură mecanică simplă. În acest caz tensorul de cuantificare entropie-flux de entropie este o funcție liniară de tensorul energie-impuls al obiectului mecanic. Legea de compoziție pentru evenimente relativiste este cunoscută, ea este aditivă în raport cu tensorii energie impuls. Este în consecință necesar să avem o lege de compoziție analogă pentru evenimentele gnostice dacă trebuie păstrată corespundența menționată pentru evenimentele compuse.

2) După cum s-a menționat mai sus, tensorii gnostici sînt tensori entropie-flux de entropie. După cum se știe, entropia termodinamică macroscopică este aditivă.

3) Pentru o incertitudine slabă ambele legi de compoziție (38) tind una spre alta și corespund atunci sumării erorilor relative patratice și compunerii aditive a acestor erori. Acest caz particular se potrivește astfel celei mai uzuale legi de compoziție din statistică.

4) În cazul unei incertitudini puternice legile de compoziție (39) conduc la formule de estimare care sînt mai robuste la date anormale decît formulele de estimare ale statisticii clasice.

Aplicații ale teoriei gnostice

Caracteristicile gnostice ale unui eșantion de date

Totul a fost pregătit cu scopul de a trece la estimare. Se va spune că o caracteristică gnostică α_n a unui eșantion de date $Z(z_o, n)$ manifestă *sensibilitatea* β, γ în raport cu al n-lea membru z_n al acestei mulțimi de date dacă

$$\lim_{z_n\to 0}\frac{\alpha_n-\alpha_{n-1}}{z_n^{-\beta}}=c \qquad \lim_{z_n\to\infty}\frac{\alpha_n-\alpha_{n-1}}{z_n^{\gamma}}=c' \tag{42}$$

(c, c' = const.)

Mărimile β și γ sînt astfel măsuri fie de sensibilitate fie de robustețe ale formulelor. În cadrul teoriei gnostice pot fi obținute caracteristici cu diferite sensibilități. Oricare astfel de caracteristici pot fi calculate utilizînd doar datele reale, fără un model statistic.

Notînd $\bar{q}$ media aritmetică $\frac{1}{n}\sum_{i}^{n} q_i$ a unei mărimi putem obține (40) în forma

$$w_q = n\sqrt{(\overline{f^{-1}})^2 - \overline{h_q^2}} \qquad w_q = n\sqrt{\bar{f}^2 + \overline{h_e^2}} \qquad (41')^*$$

și introducem noi caracteristici ale unei mulțimi de date $Z(z_o, n)$ astfel încît

$$\overline{h_q^2} = \frac{1}{n}\left(\overline{h_q^2} + 2\sum_{k=1}^{n-1}\frac{n-k}{n}C_q(k)\right)$$

$$\overline{h_e^2} = \frac{1}{n}\left(\overline{h_e^2} + 2\sum_{k=1}^{n-1}\frac{n-k}{n}C_e(k)\right) \qquad (42')^*$$

unde $\overline{h_q^2}$ și $\overline{h_e^2}$ pot fi numite *dispersia de cuantificare* și, respectiv, *de estimare*, și unde mărimile $C_q(k)$ și $C_e(k)$

$$C_q(k) := \frac{1}{n-k}\sum_{i=1}^{n-k} h_q(2\,\Omega_i)h_q(2\,\Omega_{i+k})$$

$$C_e(k) := \frac{1}{n-k} h_e(2\,\omega_i)h_e(2\,\omega_{i+k}) \qquad (43)^{**}$$

sînt generalizări gnostice ale *coeficienților de corelație*. Au loc evident

$$C_q(O) = \overline{h_q^2} C_e(O) = \overline{h_e^2} \qquad (44)^{***}$$

ca în cazul dispersiilor și coeficienților de corelație statistici obișnuiți. Toate aceste mărimi pot fi calculate ca funcții de raportul $\zeta_i = z_i/z_o$:

$$\bar{h}_q = \frac{1}{n}\sum_{i}^{n}\frac{\zeta_i^2 + \zeta_i^{-2}}{2} \qquad \bar{h}_e = \frac{-1}{n}\sum_{i}^{n}\frac{\zeta_i^2 - \zeta_i^{-2}}{\zeta_i^2 + \zeta_i^{-2}} \qquad (45)$$

$$\overline{f^{-1}} = \frac{1}{n}\sum_{i}^{n}\frac{\zeta_i^2 + \zeta_i^{-2}}{2} \qquad \bar{f} = \frac{1}{n}\sum_{i}^{n}\frac{2}{\zeta_i^2 + \zeta_i^{-2}} \qquad (46)$$

$$C_q(k) = \frac{1}{n-k}\sum_{i=1}^{n-k}\frac{\zeta_i^2 - \zeta_i^{-2}}{2}\,\frac{\zeta_{i+k}^2 - \zeta_{i+k}^{-2}}{2} \qquad (47)$$

$$C_e(k) = \frac{1}{n-k}\sum_{i=1}^{n-k}\frac{\zeta_i^2 - \zeta_i^{-2}}{\zeta_i^2 + \zeta_i^{-2}}\,\frac{\zeta_{i+k}^2 - \zeta_{i+k}^{-2}}{\zeta_{i+k}^2 + \zeta_{i+k}^{-2}}\,*$$

---

* În original formulele (41') și (42') sînt notate (41) ș,respectiv (42), numerotare folosită însă pentru formulele anterioare (N.R.).

** În original în membrii stîngi apare în mod eronat c_q (k), respectiv c_e (k) (N.R.).

*** În original membrul stîng din a doua relație e notat greșit c_e (k) (N.R.).

**** În original la numitorul ultimei fracții din formula (47) pentru C_e (k) apare în mod eronat semnul — (N.R.).

Ambele caracteristici $\overline{h_q}$ și $\overline{h_e}$ sînt legate de asimetria mulțimii de date și de o eroare sistematică. La incertitudini slabe ambele tind spre o mărime proporțională cu eroarea relativă medie. Mărimile $\overline{h_q^2}$, $\overline{h^2}$, $\overline{f^{-1}}$ și $\overline{f}$ caracterizează dispersia datelor, iar C_q cu C_e corelațiile lor. La incertitudini slabe toate acestea tind spre mărimi proporționale cu dispersiile de selecție și, respectiv, coeficienții de corelație obișnuiți. La incertitudini puternice caracteristicile gnostice diferă substanțial de cele statistice. Ele sînt fie mai robuste în raport cu datele anormale decît formulele statistice, fie mai sensibile. Aceasta permite o alegere a caracteristicilor unui eșantion de date astfel încît ele să fie adaptate cerințelor fiecărei probleme particulare. Utilizînd formule gnostice insensibile la date anormale obținem estimații robuste ale caracteristicilor unui eșantion de date. Pe de altă parte, testarea unei ipoteze asupra existenței unei date anormale într-un eșantion de date poate fi mai eficientă cînd se utilizează o formulă sensibilă.

Toate caracteristicile menționate ale unui eșantion de date sînt funcții de valoarea ideală necunoscută z_0 care trebuie să fie estimată. O estimație z_0 a mărimii z_0 este o generalizare gnostică a unui parametru de poziție al unui eșantion de date (un caz foarte particular de astfel de parametru este media aritmetică). Dependența de z_0 a tuturor caracteristicilor gnostice permite să se obțină diferite estimații ale lui z_0 prin minimizarea sau maximizarea diferitelor caracteristici.

Sensibilitatea caracteristicilor gnostice individuale este rezumată în Tabelul 3:

Tabelul 3 **Sensibilitatea caracteristicilor gnostice ale unui eșantion de date**

Cuantificare	Caracteristică	$\overline{f^{-2}}$	$\overline{h_q^2}$	$\overline{f^{-1}}$	$\overline{h_q}$	w_q	C_q
	Sensibilitate	4	4	2	2	2	2
Estimare	Caracteristică	$\overline{f^2}$	$\overline{h_e^2}$	$\overline{f}$	$\overline{h_e}$	w_e	C_e
	Sensibilitate	—4	0	—2	0	0	0

Există astfel șase generalizări gnostice diferite ale dispersiei de selecție ($\overline{f^{-2}}$, $\overline{h_q^2}$, $\overline{f^{-1}}$, $\overline{f}$, $\overline{h_e^2}$, $\overline{f^2}$) avînd cinci grade de sensibilitate de la 4 la —4. Generalizările gnostice ale erorii medii și ale covarianțelor au două modificări cu sensibilitate 0 și respectiv 2. Toate aceste caracteristici tind spre mărimi proporționale cu caracteristicile statistice obișnuite cînd datele sînt suficient de precise. Ponderile w_q și w_e tind în astfel de condiții la dimensiunea n a eșantionului de date. Într-un caz mai general ponderile relative w_q/n și w_e/n caracterizează divergența datelor.

Estimarea parametrilor de poziție ai unui eșantion de date

După cum s-a menționat mai sus, ca o estimație potimală a unui parametru de poziție al unui eșantion de date luăm o mărime $\tilde{z}_0$ satisfăcînd o condiție de optimalitate aplicată unei caracteristici gnostice. În Tabelul 4 sînt rezumate opt tipuri de astfel de condiții. Pentru a distinge estimațiile individuale $\tilde{z}_0$ introducem notația z_J (J=qI, qC, qE, qS, eS, eF, eI).

Optimalitatea estimațiilor este astfel strîns legată de extremalizarea informației, a entropiei termodinamice și a componentelor tensorilor gnostici. Estimarea efectivă tinde astfel să se apropie cît mai mult posibil de ciclul gnostic ideal.

Tabelul 4 **Condiţii de optimalitate pentru estimarea unui parametru de poziţie z_0 al unui eşantion de date**

Estimaţia $\tilde{z}_0$	Condiţia de optimalitate pentru $\tilde{z}_0$
z_qI	$\frac{d\overline{h_q^2}}{dz_0}=0$
z_qC	$\frac{d}{dz_0}\left((\overline{h}_q)^2-\frac{1}{n}\overline{h_q^2}\right)=0$
z_qF	$\frac{d\overline{f}^{-1}}{dz_0}=0$
z_qS	$\overline{h_q}(z_0)=0$
z_eS	$\overline{h_e}(z_0)=0$
z_qC	$\frac{d}{dz_0}\left((\overline{h}_e)^2-\frac{1}{n}\overline{h_e^2}\right)=0$
z_eF	$\frac{d\overline{f}}{dz_0}=0$
z_eI	$\frac{d\overline{h_e^2}}{dz_0}=0$

Unele eşantioane de date pot avea mai mult decît un parametru de poziţie. Într-un astfel de caz există „aglomerări" de date. Fiecare dintre aglomerări poate avea propriul său parametru de poziţie.

Problema unei estimări efective a unui parametru de poziţie poate fi formulată pe scurt după cum urmează: Se dă eşantionul de date $Z(z_0, n)$. Se dă tipul J al estimaţiei z_J. Să se găsească toate (etstimaţiile N.T.) $\tilde{z}_0$ satisfăcînd condiţia de optimalitate pentru z_J (Tabelul 4).

Se poate arăta că această problemă de estimare a parametrului de poziţie are aceleaşi soluţii z_J ca şi ecuaţiile

$$\text{A:}\quad z_J=\sqrt[4]{\frac{\sum_{i}^{n} f_i^m z_i^2}{\sum_{i}^{n} f_i^m z_i^{-2}}}\quad \text{sau}\quad \text{B:}\quad z_J=\sqrt[4]{\frac{\sum_{i\neq j}^{n} f_j^k f_i^m z_i^2}{\sum_{i\neq j}^{n} f_j^k f_i^m z_i^{-2}}} \tag{48}$$

(unde $f_i=2/((z_i/z_J)^2+(z_J/z_i)^2)$)

specificate de Tabelul 5:

Tabelul 5 **Specificațiile ecuațiilor pentru parametrul de poziție al unui eșantion de date**

Estimația z_J	Ecuația A/B	m	k	Sensibilitatea lui z_J^4			
				Numărător		Numitor	
				β	γ	β	γ
z_{qI}	A	—1	—	0	4	4	0
z_{qC}	B	0	—1	—2	2	2	—2
$z_{qF}=z_{qS}$	A	0	—	—2	2	2	—2
z_{eS}	A	1	—	—4	0	0	—4
z_{eC}	B	1	2	—4	0	0	—4
z_{eF}	A	2	—	—6	—2	—2	—6
z_{eI}	A	3	—	—8	—4	—4	—8

În situația incertitudinilor slabe se poate arăta că toate estimațiile z_J se apropie de media aritmetică $\bar{z}$ a datelor z_i $(i=1,\ldots,n)$. A fost elaborat un algoritm pentru un caz general (m dat și o incertitudine arbitrară a datelor). Aplicarea lui la date reale de proveniență diferită produce rezultate promițătoare.

Există estimații gnostice ale parametrului de poziție mai rezonabile decît cele șapte specificate de Tabelul 5.

Estimarea limitelor unui eșantion de date

Multe probleme practice — cum ar fi diagnosticurile atît medicale cît și tehnice, recunoașterea formelor, poziționarea sistemelor de alarmă etc. — sînt legate de estimarea limitelor unui eșantion de date. Astfel de limite nu sînt în mod necesar determinate de cel mai mic și cel mai mare membru al eșantionului. Valorile extreme pot fi anomalii, nu membri „adevărați" ai eșantionului. În cazul unui eșantion mic se poate întîmpla opusul, anume intervalul dintre limita inferioară z_L și limita superioară z_U poate fi insuficient acoperit cu date. Într-un astfel de caz atît cel mai mic cît și cel mai mare membru se pot situa adînc în interiorul intervalului. Utilizînd o abordare statistică am putea obține limitele unui eșantion de date pentru o semnificație dată dacă ar fi cunoscută funcția de distribuție de probabilitate. Totuși, un astfel de model statistic nu este întotdeauna disponibil în practică. Teoria gnostică oferă metode pentru estimarea limitelor direct din date fără un model statistic. Prima metodă se bazează pe o estimare directă a distribuției de probabilitate. Metoda considerată aici face uz de neliniaritatea și robustețea estimației gnostice a parametrului de poziție.

Să notăm prin $\tilde{z}_{n-1}$ estimația de tip z_{eF} sau x_e a parametrului de poziție z_0 al unui eșantion de date $Z(z_0, n-1)$. După extinderea acestui eșantion prin adăugarea unei noi date z_n estimația este $\tilde{z}_n$. Problema importantă este comportarea funcției $\tilde{z}_n(z_n)$. Se poate arăta că

$$\tilde{z}_n(\tilde{z}_{n-1}) = \lim_{\to 0} (\tilde{z}_n(z_n)) = \lim_{z_n \to \infty} \tilde{z}_n(z_n)) = \tilde{z}_{n-1} \qquad (49)$$

și că funcția $\tilde{z}_n(z_n)$ are un minim și un maxim în niște puncte z' și z''. Acestea împart intervalul R_+ în trei subintervale. Funcția $\tilde{z}_n(z_n)$ este monoton descrescătoare pe primul subinterval (0, z'). Ea crește pe al doilea interval (z', z'') și descrește din nou pe al treilea interval (z'', ∞). Comportarea sa este astfel „naturală" numai pe al doilea interval unde creșterea argumentului z_n mărește estimația $\tilde{z}_n$. Avem deci un motiv de a accepta a n-a dată z_n ca un membru „propriu" al eșantionului de date Z numai dacă $z' < z_n < z''$ și să luăm limitele z_L și z_U egale cu soluțiile pozitive finite ale ecuației

$$\frac{d\tilde{z}_n}{dz_n} = 0. \tag{50}$$

Se poate da o interpretare interesantă mărimilor $\tilde{z}_n(z_L)$ și $\tilde{z}_n(z_U)$ datorită inegalității

$$\tilde{z}_n(z_L) \leqslant \tilde{z}_n(z_n) \leqslant \tilde{z}_n(z_U) \qquad (\forall z_n,\ 0 < z_n < \infty) \tag{51}$$

Intervalul $[\tilde{z}_n(z_L),\ \tilde{z}_n(z_U)]$ este astfel un „interval de toleranță" al parametrului de poziție $\tilde{z}_n$. El caracterizează variabilitatea sau robustețea parametrului de poziție $\tilde{z}_n$. Avînd astfel un eșantion de date $Z(z_0, n-1)$ putem să clasificăm un candidat z_n în calitate de membru al datelor „de aceeași natură" după cum urmează:

$0 < z_n < z_L$ și $z_U < z_n < \infty$ ... rejectabil

$z_n = z_L$ și $z_n = z_U$ ... critic

$z_L < z_n < \tilde{z}_n(z_L)$ și $\tilde{z}_n(z_U) < z_n < z_U$ ... acceptabil dar necritic

$\tilde{z}_n(z_L) < z_n < \tilde{z}_n(z_U)$ ... acceptabil, tipic

$z_n = \tilde{z}_n$ ... acceptabil, caracteristic

Toate aceste intervale vor fi mai largi pentru m=2 ($\tilde{z}_n \equiv z_{eF}$) și mai înguste pentru m=3 ($\tilde{z}_n \equiv z_{eI}$, vezi Tabelul 5). O altă modificare a lărgimii tuturor acestor intervale este ușor de realizat prin introducerea unui parametru de scală pozitiv s în modelul (4) al datelor:

$$z = z_0 e^{s^{-1}} \Omega' \tag{52}$$

Toate datele din același eșantion au același parametru de scală, ele diferă numai prin parametrul Ω' care caracterizează efectul individual al incertitudinii. Toate formulele teoriei gnostice sînt exprimabile în funcție de raportul z/z_0. Scala s le modifică numai prin aceea că mărimea $(z/z_0)^s$ trebuie să fie substituită lui z/z_0. Interdependența între parametrul de scală și limitele unui eșantion de date poate fi utilizată pentru estimarea parametrului de scală. O estimație a unui parametru de scală necunoscut poate fi obținută ca soluție a ecuației (50) unde o limită dată este substituită variabilei z_n.

Estimarea probabilității

Mărimile p_e din (34) și $1-p_e$ au apărut ca parametri ai variației prin estimare a informației I_e (32). Ele joacă astfel un rol analog probabilității deși nu considerăm un model probabilistic al incertitudinii. Putem vedea că aceste mărimi au de fapt

importante însușiri ale probabilității. Atît p_e cît și $1-p_e$ sînt definite ca funcții de argumentul Ω pe intervalul $(-\infty, \infty)$. Ambele variază monoton între 0_+ și 1_-. Ambele pot fi utilizate pentru a obține măsuri finite pe mulțimi Borel pe R_+ (cînd se dă o dată particulară z_i). Intervalele $(0, z_i/z_0)$, $(z_i/z_0, \infty)$ au astfel lungimile p_{ei} și, respectiv $1-p_{ei}$. Cunoscînd o dată z_i putem aprecia probabilitatea ca mărimea necunoscută z_0 să depășească valoarea z_i ca p_{ei}. Acesta este un analog al probabilității $\Pr(z_0 > z_i)$ (z_i dat). Această analogie este de importanță practică prin aplicarea la estimarea probabilității unui eveniment aleator deoarece există transformări ale rezultatelor experimentelor aleatoare în datele z_i și z_0 astfel încît pot fi obținute estimațiile gnostice ale probabilității bazate pe măsuri p_{ei}. Aceasta poate fi interpretată ca o promisiune de a îmbunăți atît economia datelor cît și fiabilitatea deciziilor bazate pe eșantioane mici de date.

Estimarea directă a distribuțiilor de probabilitate din eșantioane mici de date

Se poate arăta ușor că funcția de distribuție $p_e(\Omega)$ are o funcție de densitate

$$\frac{dp_e}{d\Omega} = \frac{4}{(e2\Omega + e^{-}2\Omega)^2} = f^2 \qquad (53)$$

care poate fi rescrisă (utilizînd modelul (52)) ca o funcție de raportul z/z_0

$$\frac{dp_e}{d\Omega} = \frac{4}{((z/z_0)^{2s} + (z_0/z)^{2s})^2} \qquad (54)$$

Această expresie poate fi interpretată ca o „funcție de influență" a unei date z a cărei valoare ideală este z_0. Ea caracterizează influența evenimentului „a fost obținut rezultatul z al cuantificării lui z_0" asupra estimației noastre a posteriori a densității de probabilitate a unui alt rezultat posibil din vecinătatea lui z. „Lărgimea" acestei funcții de influență este dependentă de parametrul de scală s așa cum ne-am putea aștepta.

Din (25), (26), (32), (34) și (53) rezultă direct o relație interesantă

$$\frac{dp_e}{d\Omega'} = \frac{d^2 I_q}{dh_q^2} \qquad (55)$$

Densitatea de probabilitate egalează astfel densitatea surselor cîmpului de informație $I_q(h_q)$.

Toate egalitățile (34), (53)—(54) sînt legate de o singură dată parametrizată de parametrul Ω. Dar cunoaștem deja legea de compoziție a datelor (38) care arată două modalități de reprezentare a eșantionului de date $Z(z_0, n)$ prin evenimentele compuse parametrizate de Ω_c și ω_c. Obținem deci două distribuții de probabilitate diferite $(p_e)_q$ și $(p_e)_e$ parametrizate de Ω_{cq} și respectiv Ω_{ce}. Utilizînd acești parametri Ω ai evenimentelor compuse interpretăm egalitățile (34), (53)—(54) drept caracteristici ale întregului eșantion Z. Aceste caracteristici de tip „q" diferă de cele de tip „e" prin alegerea între versiunile de cuantificare și de estimare ale legii de compoziție (38): Mărimea Ω_{cq} este identică cu parametrul Ω_c al tensorului $K_q(2\Omega_c)$ care apare în versiunea de cuantificare a lui (38). Mărimea Ω_{ce} este o soluție a ecuației

$$\operatorname{th} \Omega_{ce} = -\operatorname{tg} \omega_c \qquad (56)$$

unde ω_c este parametrul tensorului $K_e(2\omega_c)$ care apare în versiunea de estimare a lui (38). Valorile Ω_{cq} și Ω_{ce} pot diferi substanțial și funcțiile de distribuție corespunzătoare manifestă sensibilitate/robustețe diferită în raport cu datele anormale.

Distribuțiile gnostice pot fi ușor evaluate numeric ca funcții de valoarea ideală necunoscută z_0 pentru fiecare eșantion de date, chiar de dimensiune mică, dacă este dat parametrul de scală s. După cum s-a menționat mai sus, parametrul de scală poate fi de asemenea estimat din date. Rezultă astfel că avem nevoie numai de datele reale cînd utilizăm această abordare pentru identificarea experimentală a unui model probabilistic.

Testele algoritmilor bazați pe noțiunea gnostică de densitate de probabilitate efectuate în domeniul dinamicii fiziologice au dat rezultate apreciate de specialiștii domeniului ca atît noi cît și interesante.

Identificarea robustă a unui model ARMA

Ecuațiile pentru estimarea parametrilor modelelor liniare din date contaminate de zgomot sînt cunoscute a fi liniare numai în cazul zgomotului gaussian. Soluția de verosimilitate maximă a acestei probleme pentru alte distribuții este neliniară în raport cu datele. Aceasta se asociază de asemenea cu soluțiile obținute în teoria statistică robustă. Forma particulară a acestei neliniarități depinde în aceste cazuri de funcția de distribuție a zgomotului și de variațiile ei posibile. Teoria gnostică oferă soluții robuste ale acestei probleme de identificare fără nici o ipoteză asupra modelului statistic al zgomotului.

Ideea este de a utiliza o caracteristică gnostică a discrepanței între data efectivă și valoarea ei prezisă calculată cu ajutorul unei estimații a priori a modelului identificat. Estimația a posteriori a acestui model se obține apoi prin extremalizarea caracteristicii gnostice. În acest scop este avantajoasă o soluție recursivă pentru cel puțin trei motive:

1) Volumul de calcule necesare este minimizat pentru a facilita aplicațiile în timp real.

2) Fiecărei ecuații a sistemului i se poate atribui ușor o pondere proprie determinată de incertitudinea estimată a datei corespunzătoare pentru a mări robustețea în raport cu datele anormale.

3) Se poate introduce ușor uitare exponențială pentru a obține un algoritm adaptiv utilizabil în situații nestaționare.

Astfel de algoritmi au fost elaborați și testați. Ei au operat la fel de eficient ca și algoritmul celor mai mici patrate cînd erorile datelor datorate zgomotului erau suficient de mici. La o mărire extremă a erorilor algoritmii gnostici au fost aplicabili chiar după eșecul total al versiunii clasice. Aplicabilitatea practică a unui algoritm gnostic de acest tip a fost demonstrată cu succes cu date reale pentru predicția consumului de gaz într-un sistem de distribuție a gazului.

Această soluție a problemei identificării deschide noi căi posibile pentru soluții robuste a unor probleme înrudite cum ar fi netezirea, filtrarea și analiza în timp real a seriilor de timp.

Concluzii

Teoria gnostică a datelor reale oferă noi formule ale caracteristicilor eșantioanelor de date. Aceste formule sînt direct calculabile din date. Ele includ generalizări ale caracteristicilor statistice clasice ale eșantioanelor de date și posedă fie robustețe sporită fie sensibilitate mărită în raport cu datele anormale sau normale. Sînt disponibile de asemenea noi caracteristici ale datelor individuale și ale eșantioanelor mici de date cum ar fi informația, entropia termodinamică și distribuția de probabilitate. Ele nu impun o ipoteză a priori asupra unui model statistic deoarece se

bazează numai pe date. Utilizînd caracteristicile gnostice ca funcţii criteriale pentru optimizare este posibil să se obţină noi algoritmi eficienţi şi robuşti pentru identificarea modelelor, pentru filtrare şi predicţie.

Referinţe bibliografice

[1] Kovanic, P. (1984a). Gnostical theory of individual data. *Problems of Control and Information Theory*, în curs de apariţie.
[2] Kovanic, P. (1948b). Gnostical theory of small samples of real data. *Problems of Control and Information Theory*, în curs de apariţie.
[3] Kovanic, P. (1948c). On relations between information and physics. *Problems of Control and Information Theory*, în curs de apariţie.
[4] Jumarie, G. A. (1975). A relativistic information theory model for general systems. *Int. Journal of System Science*, *6*, 865—886.
[5] Jumarie, G. (1980). *Subjectivité, Information, Système. Synthese pour une cybernétique relativiste*, Les Editions Univers Inc., Québec.
[6] Perez, A. (1957). Sur la théorie de information dans le cas d'un alphabet abstrait. *Transactions of the first Prague conference on inf. theory, stat, decision functions, random processes*, Publishing house of the Cz. Ac. of Sc., Prague, 209—243.

(continuă de la pag. 43)

— „BASIC pentru începători, cu calculatorul personal" un manual practic din ciclul „CALCULATOARE PERSONALE ŞI PROGRAMAREA LOR".

— Ciclul „PROIECTAREA ASISTATĂ DE CALCULATOR" reprezentat prin articole de direcţionare în domeniu şi articole prezentate la o primă sesiune naţională.

— Microcalculatoarele personale româneşti, Student-HC 80, PRAE (pentru acesta şi limbajul său BASIC) şi microcalculatorul profesional-personal românesc Felix PC, în prezentări sintetice — în premieră într-o carte.

În trimestrul IV 1985 apar şi volumele AMC 52—53—54, cu ciclurile amintite, dar şi cu automatizare flexibilă, roboţii, limbajul BASIC pentru WANG VS, ghidul analistului (continuare la AMC 45—46), jocuri de întreprindere.

Preţul unui volum AMC este de aproximativ 25 lei.

A. Békéssy, J. Demetrovics, I. Horváth Gaudi, L. Hannak

Institutul pentru tehnică de calcul şi automatizări, Academia de ştiinţe a R.P.U. (SZTAKI), Ungaria.

G. Buvár, G. Balogh

Centrul pentru recolte agricole (KITE), Nádudvar, Ungaria.

Rezumat. Lucrarea prezintă rezultatul cooperării între SZTAKI şi KITE. Este inclusă o evaluare statistică a rezultatelor producţiei în diferite condiţii ecologice ale mediului ambiant, planificarea tehnologiei şi controlul stocurilor.

Activitatea denumită „planificarea tehnologiei" este prezentată cu mai multe detalii.

Cuvinte-cheie. Agricultură, aplicaţii ale tehnicii de calcul; prelucrare a datelor; programare matematică, optimizare, gestiune şi control al stocurilor.

Introducere. Ce este KITE?

În cele ce urmează vom prezenta rezultatele cooperării dintre două institute maghiare, primul fiind Institutul pentru tehnică de calcul şi automatizări al Academiei de ştiinţe a R.P.U. (SZTAKI), iar al doilea este KITE. Aceasta este prescurtarea în limba maghiară a numelui „Cooperarea în producţie pentru porumb şi culturi industriale". Numele indică faptul că este vorba de o organizaţie agricolă, dar KITE este un cuvînt cheie în agricultura ungară şi de asemenea în lucrarea de faţă, astfel încît considerăm justificate cîteva rînduri în care să v-o prezentăm.

În ţara noastră, asemănător altor cîteva ţări, agricultura s-a transformat într-un fel de *industrie*. Această transformare se datorează parţial colectivizării micilor ferme, dar chiar şi cele necolectivizate au devenit asemănătoare unor ateliere industriale, prin două aspecte importante: ferma modernă are nevoie de forţă de muncă, iar rezultatul producţiei este vîndut pe piaţă, în loc să acopere numai nevoile familiei. De asemenea, dacă acum 40 de ani, a fi ţăran, însemna să aparţii unei clase sociale, de nivel mai degrabă coborît, în prezent aceasta înseamnă pur şi simplu o ocupaţie sau chiar o profesie care poate fi profitabilă dacă este bine exercitată.

Transformarea societăţii nu s-a putut realiza fără depăşirea unor dificultăţi serioase.

Deşi competenţa şi instrumentaţia agricolă corespunzătoare producţiei pe scară mică erau tradiţionale la sate, metodele agriculturii moderne nu erau cu adevărat cunoscute, sau chiar cunoscute fiind nu erau accesibile înainte de colectivizare şi chiar un anumit timp după aceasta.

În prezent producţia pe scară largă în fermele colectivizate — cooperativele agricole — este dirijată de aşa-numitele sisteme de producţie agricolă. Există cîteva astfel de sisteme aplicate cu succes diferitelor culturi (de exemplu porumb), dar două sisteme sînt de o importanţă covîrşitoare pentru ţara noastră, iar unul dintre acestea este reprezentat de KITE. Activitatea desfăşurată de SZTAKI referitor la automatizarea în agricultură este legată de obiectivele şi problemele KITE.

KITE a fost înfiinţată în anul 1972, cînd reprezentanţii a nouă cooperative agricole au hotărît că trebuie făcut ceva pentru modernizarea agriculturii ungare şi că trebuie început cu cele mai importante culturi.

Cartierul general o fost stabilit la Nádudvar, un mic sat, nu departe de Debreţin unde cooperativa „Vörös Csillag“ (Steaua roşie) a găsit loc pentru noua organizaţie creindu-i condiţii pentru a-şi începe activitatea.

Între timp 397 de cooperative au hotărît să se ralieze la KITE, în sensul acceptării şi aplicării sistemelor de producţie oferite pentru mai multe culturi (de exemplu: porumb, grîu, floarea soarelui, sfeclă de zahăr, orez etc.). Ralierea trebuie înţeleasă într-un sens special: aşa-numitele *ferme-membre* îşi păstrează deplina independenţă. Ele decid ca şi înainte de raliere, ce doresc să producă şi în ce cantitate. Relaţiile lor cu KITE sînt cele dintre un producător şi un serviciu de consultanţă: ele solicită serviciile KITE, le plătesc şi astfel asigură menţinerea organizaţiei. Cu toate acestea, pentru o exprimare concisă, vom utiliza denumirea de fermă-membru. De asemenea KITE este independentă deoarece, deşi este coordonată (supervizată) atît de stat cît şi de fermele-membri, nu este finanţată de către stat. Suprafeţele cultivate de fermele-membre ocupă o treime din totalul suprafeţei arabile a Ungariei.

S-a menţionat deja că lipsa de cunoaştere şi de disponibilitate a tehnologiilor agricole moderne a stat la originea principalelor dificultăţi ale fermelor cooperativizate. Ca urmare, KITE, asemănător altor organizaţii similare, îşi desfăşoară activitatea în două direcţii esenţiale:

ca, consultant:

— colectează informaţii referitoare la rezultatele relevante, tehnice şi ştiinţifice, obţinute în întreaga lume;

— analizează şi testează rezultatele înaintea aplicării generalizate;

— adaptează rezultatele la condiţiile locale;

— evaluează rezultatele tehnologiilor recomandate;

— perfecţionează tehnologiile pe baza experienţei practice.

ca partener:

— oferă maşini, *servicii* de reparaţii şi *piese de schimb*, precum şi

— *seminţe, fertilizatori, chimicale.*

Fermele-membri îşi cheltuiesc banii proprii pentru serviciile KITE, urmărind cu atenţie dacă investiţia este profitabilă sau nu.

De altfel, relaţia dintre SZTAKI şi KITE este bazată pe aceleaşi principii, respectiv pe interesul reciproc. Serviciile SZTAKI înseamnă cheltuieli pentru KITE, deci SZTAKI trebuie să se facă util în această relaţie, în caz contrar nemaifiind posibilă investirea altor bani pentru cooperare. Totuşi relaţia noastră poate fi considerată cu adevărat astfel (cooperare) deoarece nu numai membrii responsabili din SZTAKI, ci şi cei din KITE trebuie să muncească din greu pentru obţinerea succeselor.

Analiza rezultatelor producţiei agricole

Aşa cum s-a menţionat deja, KITE urmăreşte în permanenţă situaţia tehnologiilor sugerate şi acceptate. Un sistem de informare regulată este în funcţiune din anul 1974; acesta constituie un set de date de bază. Acest set conţine aproximativ patruzeci de caracteristici agrotehnice şi ecologice de bază referitoare la 5 000 de zone de producţie (cîmpuri) pe care se produc regulat grîu, porumb şi floarea soarelui. Cooperarea noastră a început în anul 1976 prin analiza rezultatelor producţiei pe aceste cîmpuri, iar noi am căutat conexiunile semnificative dintre particularităţile tehnologiei aplicate (de exemplu: sortimentul de seminţe, cantitatea şi calitatea fertilizatorilor etc.) şi rezultatele obţinute *în diferite condiţii ecologice*. Prelucrarea se efectuează în fiecare an începînd din 1976. A fost aplicată analiza statistică multivariabilă şi sîntem în situaţia fericită în care seturile noastre de date sînt suficient de mari. Chiar dacă munca prealabilă efectuată asupra datelor reprezintă o corvoadă neplăcută, ca statisticieni trebuie să fim mulţumiţi de seturile noastre mari de date. După aceşti ani de muncă conştiincioasă am reuşit să ne clarificăm în cîteva probleme importante; avem în prezent la dispoziţie metode prin care putem estima cu o mare probabilitate, ce paşi ai unei tehnologii propuse pot avea o influenţă considerabilă în anumite condiţii ecologice, asupra rezultatelor producţiei. Aceste rezultate sînt integrate în prezent în recomandările tehnologice ale KITE.

Planificarea tehnologiei: o prezentare

În perioada în care producţia agriculturii ungare era insuficientă, majoritatea eforturilor au fost dirijate spre propagarea şi difuzarea metodelor care pot asigura creşterea producţiei.

Peste o anumită limită însă, costurile de producţie cresc mai rapid decît producţia, astfel încît interesul agronomilor a început să se orienteze şi spre problemele economice. Astfel, s-au făcut tentative pentru găsirea, prin metode matematice, a unui model mai mult sau mai puţin general pentru o structură optimă a producţiei. Cu toate acestea, rezultatele nu au fost încă aplicate în practică.

Cauza posibilă a acestui eşec (dacă poate fi denumit eşec) trebuie căutată probabil în faptul că scopul acestor eforturi l-a constituit determinarea, în general, a planului optim al structurii culturilor pentru o fermă. Dar fiecare fermă are propria sa structură, iar transformarea unei stueturi existente este foarte dificilă. Dacă, de exemplu, structura optimă presupune cultivarea cu soia a unui anumit cîmp, iar acest tip de plantă nu a fost niciodată produs înainte, atunci lipsa cunoştinţelor şi maşinilor necesare vor face imposibilă realizarea planului. De asemenea, în anumite părţi ale Ungariei, unele culturi au fost produse în mod tradiţional timp de mai multe decenii, oamenii de acolo nu sînt tentaţi să încerce ceva nou şi probabil că nici din punct de vedere al interesului public nu ar fi deloc indicat să o facă (cu totul altul este desigur cazul în care introducerea unei noi culturi este iniţiată de însăşi ferma respectivă). Transformarea şi dezvoltarea viitoare a unei culturi existente au reguli bine determinate şi cunoscute, prin care agronomii locali pot găsi, după luarea în considerare a factorilor relevanţi (economici şi alţii) alternative fezabile din punct de vedere al structurii producţiei.

KITE are o aşa-numită *tehnologie-cadru* pentru 47 de tipuri de culturi.

Aceasta este o listă lungă de operaţii (100—300), fiecare dintre acestea fiind un pas a cărui execuţie este recomandată ca obligatorie sau opţională. Tot aici este descris pentru fiecare operaţie, necesarul de resurse respectiv maşini de putere şi alte implementări, materiale etc. Procedura normală de planificare începe cu o întîlnire între reprezentanţii KITE (care au în bagajele lor tehnologiile-cadru) şi aceia ai fermelor-membri care cunosc, cel puţin în principiu ce doresc să semene, în ce cantitate şi unde. Rezultatul unei astfel de întîlniri este un lung raport comun, o înţelegere în care este descrisă, de data aceasta detaliat, pentru fiecare lot al fermei, tehnologia asupra căreia s-a căzut de acord. Acestea sînt denumite *variante ale tehnologiei cadru*, şi constau în seturi selectate din tehnologia cadru, adaptate acum la loturile luate în considerare.

Variantele trebuie să fie mai mult sau mai puţin fezabile din punct de vedere al fermelor-membri ale căror resurse nu sînt de loc nelimitate.

Este posibil ca, între variantele tehnologiei-cadru să apară operaţii care nu există în tehnologia cadru generală, rămase din perioade în care unele ferme au decis să experimenteze metode care nu au fost sugerate de KITE (de exemplu irigaţia porumbului). Există de asemenea culturi pentru care KITE nu are o tehnologie cadru. Este la latitudinea fermelor să abordeze totuşi aceste culturi, iar etapele planului tehnologic stabilit în acest caz vor fi încadrate de asemenea între variantele tehnologiei cadru.

Fiecare etapă tehnologică de pe listă implică cheltuieli. În prezent utilizarea maşinilor de putere este cel mai costisitor factor în producţia agricolă ungară. Ca urmare, exploatarea optimă a acestora este crucială din punctul de vedere economic. De asemenea fiecare etapă de pe listă are limitele sale în timp. Două date calendaristice sînt fixate în tehnologia cadru ca recomandări pentru începutul şi sfîrşitul lucrării.

Se poate vedea acum că variantele tehnologice, adaptate masiv la condiţiile fermei, determină aproape totul. Totuşi există încă unele mici grade de libertate în care tehnicile matematice şi de calcul pot să intervină: mai întîi, tehnologiile cadru, chiar variantele selecţionate şi adaptate permit încă alternative din punct de vedere al configuraţiei maşinilor; în al doilea rînd, în intervalul de timp specificat, fermierii

sînt liberi să aleagă momentul în care doresc să treacă la execuția operațiilor specificate. Costurile de producție pot fi reduse prin selecționarea celor mai bune alternative și prin planificarea lor în timp.

Aceasta înseamnă optimizare într-un anumit sens. Dacă planul este executabil pentru fermă fără nici o dificultate, nu mai rămîne oricum nimic de optimizat în continuare. Pe de altă parte, dacă planul este departe de a fi realizabil atunci de asemenea nu rămîne nimic de reconsiderat din ansamblul situației. Practica însă a demonstrat că planurile sînt de obicei ușor supraîncărcate, astfel încît resursele fermei nu sînt chiar suficiente. De fapt noi asigurăm o reducere a lipsei de capacitate în anumite perioade ale anului, prin planificarea în timp și proiectarea celor mai bune configurații de mașini posibile, din punct de vedere al mașinilor de putere (care așa cum s-a menționat sînt factorul cel mai costisitor).

Dacă supraîncărcarea rezultată nu este catastrofică ferma poate găsi soluții, de exemplu să închirieze mașini sau să mărească timpul de lucru. De asemenea este la latitudinea fermei să cumpere noi mașini, eventual de la KITE. Dar toți acești pași vor duce pe de altă parte la creșterea costurilor de producție.

Ca *rezultat al calculelor noastre*, planificarea în timp a variantelor tehnologice și configurațiile mașinilor sînt oferite împreună cu diagrame referitoare la cost și profit. Acestea din urmă pot fi determinate de asemenea deoarece experiența dobîndită de KITE face posibilă estimarea rezultatelor previzibile în condiții normale, presupunînd că tehnologia planificată va fi respectată cu strictețe. Aici este sfîrșitul prezentării. Considerăm că a fost suficientă pentru înțelegerea caracteristicilor esențiale ale ceea ce noi denumim planificare și tehnologie.

Din considerente legate de concizia și claritatea exprimării acestea au fost prezentate într-o formă simplificată: realitatea este, ca de obicei, mai complicată. Vom continua discuția trecînd la detalii matematice, așa cum fac toți matematicienii, înșirînd definiții și formule.

Planificarea tehnologiei: modelare și matematică

Intrare. Un set selecționat de operații dintr-o listă — tehnologie cadru poate fi văzut în fig. 1.

Prima coloană conține codul unei operații a cărei denumire (prescurtare în limba maghiară) se găsește în coloana a doua. O variantă tehnologică pare similară, dar conține un număr de elemente selecționate din tehnologia cadru generală, este legată de un anumit cîmp al unei ferme, iar operațiile sînt renumerotate și listate în ordinea în care trebuie executate.

Ceea ce nu poate fi văzut în fig. 2 este faptul că, în plus, există anumite reguli suplimentare corespunzător cărora unele operații trebuie executate simultan. Dar toate aceste reguli sînt de asemenea memorate în calculator.

Apoi, există două date calendaristice indicînd intervalul de timp sugerat pentru respectiva operație. Urmează descrierea a cel mult trei configurații de mașini, propuse. O configurație constă dintr-o mașină de putere și cel mult două mașini speciale (implementări construite pentru executarea acestei lucrări). Sînt indicate codurile de grup ale mașinilor: mașini din aceeași categorie (grup) dar provenind de la fabrici diferite pot avea eficiențe diferite, astfel încît un tip, în general cel mai bun, din fiecare grup este considerat standard, iar eficiența celorlalte este măsurată procentual în raport cu cea a tipului standard. Astfel a patra coloană din secțiunile configurațiilor de mașini se referă la tipul standard de mașini și la o oră de lucru, în unități indicate în a treia coloană. Necesarul de operatori pentru mașini este implicit, dar unele lucrări solicită de asemenea forță de muncă suplimentară, și într-un astfel de caz aceasta este indicată în ultima coloană a fiecărei configurații de mașini propusă.

În calculator sînt memorate liste referitoare la toate mașinile existente la ferme, date despre eficiența acestora, costurile de utilizare și prețurile de achiziționare care determină amortizarea. De asemenea în calculator sînt memorate liste la fel de detaliate referitoare la chimicalele pentru protecția plantelor și fertilizatorii artifi-

1. *KERETTECHNOLOGIA*

(tehnologie cadru)

ETK. ÖSZI BÚZA,	*TAK. ÖSZI BÚZA*	*ÖSZI ÁRPA,*	*ROZS*
(grîu de toamnă)	(grîu, nutreț)	(orz)	(orez)

(Lucrări pregătitoare, vara)

(No.)	(Oper.)	(Unit)	(Timp)	(CONFIGURAȚIA 1)					(CONFIGURAȚIA 2)					(CONFIGURAȚIA 3)				
				(PM)	(SP1)	(SP2)	(P)	(M)	(PM)	(SP1)	(SP2)	(P)	(M)	(PM)	(SP1)	(SP2)	(P)	(M)
1.	HORCS IRTAS	HA	JUL1—AUG1				0.1	1										
2.	TARLO HANTS	HA	JUL1—AUG1	NTR	NTC2	—	4.0	—	NU	NTC1	—	3.5	—	NL	NTC1	—	2.0	—
11.	MT RAK SZAL	T	AUG1—SZE3	SL2	—	—	10.0	—	SL1	—	—	10.0	2					
12.	MT SZALLITS	T	AUG1—SZE3	TA1	KPK	—	10.0	—	KU	KPK	—	5.0	—					
17.	SZANTS ZOCM	HA	AUG1—SZE3	NTR	EKE4	—	2.0		NU	EKE3	—	1.4	—	NL	EKE3	—	0.8	—

Fig. 1. Selecțiuni dintr-o listă tehnologie-cadru a KITE.

ciali. „Parametrii" acestora influențează gradul de încărcare al forței de muncă și calculele de cost. Aceste tabele, împreună cu cele care cuprind prețurile curente de vînzare a produselor formează datele de bază, comune tuturor calculelor.

Datele particulare de intrare referitoare la o fermă constau din lista mașinilor de acolo și o descriere a cîmpurilor (dimensiune, valoare relativă etc.) ca date de bază precum și din variantele tehnologice codificate, cu un număr total de 3 000—6 000 de operații asociate planului de structură pentru o singură cultură. (De obicei nu este prezentat spre analiză un singur plan ci mai multe, ca posibile alternative.)

La sfîrșit există întotdeauna unele „observații" speciale sau condiții ca, de exemplu: „Sfecla de zahăr va fi transportată la fabrică cu mijloacele de transport ale fabricii" sau „Loturile particulare ale cooperatorilor vor fi arate cu mașinile cooperativei" care trebuie luate într-un fel în considerare în calcule dacă acest lucru este posibil (și dacă au fost comunicate de la început și nu după ce s-a constatat că rezultatele calculelor sînt eronate).

Algoritmi. Fie K numărul total de operații din lista variantelor tehnologice adaptate pentru un cadru. Fie k numărul de ordine al unei operații (k=1, 2, ..., K). Să notăm cu J numărul total de configurații de mașini recomandate ($J \leqslant 3$).

Ca unitate de timp a fost ales un interval de zece zile. O astfel de perioadă este numită „dekád" în limba maghiară și va fi denumită în continuare prescurtat TD (Ten Days — zece zile). TD-urile sînt ajustate conform lunilor calendarului, astfel încît fiecare lună este formată din trei TD-uri. Din această cauză unele TD-uri sînt mai lungi sau mai scurte decît zece zile, dar acest lucru este luat în considerare. Fie t_k^1 numărul de ordine al primului TD din intervalul de timp prescris pentru operația k, adică momentul propus pentru începerea operației și, similar, t_k^2 ultimul TD, propus pentru încheiere.

Pornind de la dimensiunile cîmpului (și de la alți parametri ai acestuia) trebuie calculat volumul de muncă necesar pentru efectuarea operației k, măsurat în ore de lucru. Fie x_{ijk} notația pentru numărul de ore necesare operației k în TD-ul i, utilizînd configurația j ($J=j_k^1, j_k^2, j_k^3$). Rezultatele activității sînt măsurate în unități diferite pentru operații diferite (hectare pentru unele activități, tone pentru altele) astfel încît dacă q_{ik} este măsura eficienței pentru configurația j, atunci se va utiliza măsura relativă:

$$p_{jk}=q_{jk}/M_k$$

unde M_k este totalul resurselor necesare pentru operația k.

Deci necesarul de muncă este satisfăcut dacă egalitățile:

$$\sum_{=t_k^1}^{t_k^2} \sum_{j=j_k^1}^{j=j_k^3} p_{jk} \cdot x_{ijk}=1, \qquad (k=1, 2, \ldots, K)$$

sînt valabile.

Unele operații pot fi executate *numai în serie* (unele după altele).

Dacă intervalele de timp prescrise pentru ele se suprapun, atunci

$$\sum_{i=t_1}^{t_1+l} \sum_{j} p_j k_2 x_{ij} k_2 \leqslant \sum_{i=t_1}^{t_1+l} \sum_{j} p_{jk'} x_{ijk'}$$

$$(l=0, 1, \ldots, t_2—t_1—1)$$

sînt inegalitățile care exprimă această condiție. Un set similar de formule exprimă faptul că unele operații trebuie efectuate simultan.

Fie b_{ij} capacitatea celei de a j-a mașini de putere în cel de al i-lea TD. Aproape toate fermele posedă mașini proprii de putere de tipul j; în acest caz, b_{ij} exprimă capacitatea *totală* a acestora, măsurată în ore.

Dacă capacitățile (resursele) acoperă încărcările, atunci inegalitatea

$$\sum_k x_{ijk} \leqslant h_{ij} \text{ ar fi valabilă pentru orice i, j.}$$

Dar acest lucru nu se întîmplă aproape niciodată. Să notăm deci z_{ij} încărcarea suplimentară față de capacitate

$$\forall_{i,j} : \sum_k x_{ijk} = b_{ij} + z_{ij}$$

Din setul tuturor soluțiilor cu x_{ijk} satisfăcînd condițiile menționate mai sus dorim să o selectăm pe aceea pentru care

$$\max_{i,j} \; (|z_{ij}|/b_{ij}) \text{ este minimal.}$$

Evident modelul, așa cum se găsește, nu va maximiza profitul. Nu a fost, de exemplu, luat în considerare faptul că pentru configurații de mașini, diferite dar la fel de acceptabile, pot să apară cheltuieli diferite. Totuși acest lucru nu este foarte important. Apariția unor deficiențe (relativ) grave de capacitate reprezintă situația cea mai defavorabilă care ar trebui evitată cu orice preț.

Vorbind despre deficiențe, la o privire mai atentă se poate observa că modelul, așa cum a fost prezentat anterior nu este deloc satisfăcător. Dacă

$$\max \; (|z_{ij}|/b_{ij})$$

este minimizat, atunci atît supraîncărcările de capacitate ($z_{ij} > 0$) cît și subîncărcările ($z_{ij} < 0$) vor fi considerate echivalente în timp ce în cazul nostru supraîncărcările sînt mult mai grave. Ca urmare, ar fi probabil mai bine să minimizăm $\max (z_{ij}/b_{ij})$, dar numai pentru indicii i, j cu $z_{ij} > 0$. (Pentru încărcări diferite dar acceptabile, acești indici pot fi, în general, diferiți.)

Nu știm dacă acest tip de model a fost analizat vreodată. De aceea, începînd din acest punct, am urmat două direcții.

Una dintre acestea a constat în abordarea *liniară*. Ca și pentru funcțiile obiectiv de (minimizat) a fost aleasă expresia:

$$\sum_{ij} \left(z_{ij} - \sum_k x_{ijk} \right) / b_{ij}$$

cu condițiile suplimentare

$$\forall_{j,i} : \sum_k x_{ijk} \leqslant b_{ij} + z_{ij}, z_{ij} \geqslant 0$$

Această problemă poate fi rezolvată utilizînd pachete de programe standard.

Din experiența noastră însă, calculatoarele de care dispunem în prezent sînt prea mici pentru a face față acestor calcule, numărul condițiilor restrictive fiind în general prea mare ceea ce conduce la durate de calcul prea mari chiar în condițiile în care capacitatea de memorie internă este suficientă. Cealaltă direcție pe care am urmat-o s-a bazat pe unele aproximații, conducînd spre o soluție de lucru care poate fi găsită întotdeauna și este mai rapidă. Această abordare este cu atît mai importantă cu cît noi dorim ca algoritmii noștri să fie prelucrați pe calculatoarele mici, personale, care vor fi instalate într-un viitor apropiat chiar la fermele-membre.

Din dorința de a simplifica problema, am introdus două condiții suplimentare: (1) fiecare operație trebuie executată, de la început pînă la sfîrșit de același tip de configurație de mașini; dacă, de exemplu, operației îi este atașată prima configurație

în primul TD permis, atunci aceeași ar trebui să execute lucrarea în toate TD-urile următoare, și (2) fiecare lucrare trebuie efectuată continuu și cu intensitate constantă.

Prin urmare, algoritmul nostru simplu, aproximativ, pornește de la prezumția că munca este distribuită egal în toate intervalele permise de timp și că prima configurație de mașini sugerată lucrează în continuu și peste tot. Ca urmare, în anumite perioade critice, pare să se manifeste o însemnată criză de capacitate.

Lipsa de capacitate în aceste perioade poate fi diminuată prin împingerea încărcării unor operații spre alte TD-uri permise. Apoi algoritmul încearcă să reducă și supraîncărcarea reziduală prin schimbarea configurației de mașini pentru unele operații, în perioadele critice. Și aceasta este tot evident.

Condițiile menționate anterior trebuie să fie satisfăcute. De asemenea este mai mult decît probabil că algoritmul nu va funcționa de loc acolo unde planurile generale de structură a culturii și tehnologiile adaptate nu sînt corelate în prealabil cel puțin la modul aproximativ cu resursele fermei. Cu toate acestea, metoda este utilizată din 1982 și a dat rezultate, dacă nu pentru toate fermele, atunci pentru cele mai bine organizate dintre ele. Este rapidă și deci mai puțin costisitoare, iar rezultatele ei pot fi comparate, pentru fermele mici, cu cele obținute prin metode „mai exacte".

Planificarea este urmată de un calcul al costurilor și profiturilor. Costul producției este calculat, chiar în eventualitatea în care rămîn unele încărcări neacoperite, ca și cum acestea nu ar exista (sau, cu alte cuvinte ca și cum mașinile de putere ar putea lucra mai mult decît este normal, ceea ce de fapt acestea pot să facă pînă la o anumită limită), dar fermele trebuie să fie pregătite să plătească un preț mai mare pentru orele suplimentare. Sau, este posibil ca fermele să decidă achiziționarea unor noi mașini. Probabil că ele au făcut deja acest lucru, iar ceea ce rezultă din calculele noastre este, în esență un răspuns la importanta întrebare: cîte mașini sînt, în mod inevitabil, necesare, pentru a face ca planul să fie fezabil.

În general, 3 din 4 alternative de structură de plan de cultură sînt prezentate pentru prelucrare, după care fermele membre decid care dintre alternative este preferabilă.

În această alegere se folosesc de un set de tabele.

Ieșire. încărcarea mașinilor de putere de tipul TA1 (este un anumit tip de autocamion) pentru o fermă care posedă 11 astfel de autocamioane ar putea constitui un exemplu. Planificarea a pus în evidență o lipsă de capacitate, mai întîi în cel de-al 28-lea TD (începutul lui octombrie) cînd încărcarea este de 1 961 ore de lucru, disponibile fiind numai 1 758 ore. Diferența este de 203 ore distribuite pe parcursul a 10 zile. Aceasta înseamnă 2 ore suplimentare de lucru pe zi pentru fiecare autocamion. Ferma nu va cumpăra noi autocamioane numai pentru acoperirea acestei supraîncărcări.

Pentru toate tipurile de mașini de putere (autocamioane, combine-recoltatoare, tractoare) au fost întocmite diagrame similare. În afara diagramelor, încărcările sînt listate de asemenea în tabele numerice, subîmpărțite pe cîmpuri. Astfel, dacă apare o supraîncărcare substanțială pe o diagramă, este ușor de depistat originea acesteia, respectiv tipul de cultură pe care ferma este pe cale să-l producă în exces față de resursele sale prezente.

Tabelele referitoare la costurile de producție pe tipuri de culturi precum și valorile estimate ale profitului aparțin de asemenea ieșirilor. La sfîrșit (sau mai bine la început) vin mesajele de eroare.

Planificarea tehnologiei, așa cum a fost prezentată se referă la un an întreg. Deoarece factorii meteorologici sau alte evenimente semnificative pot să influențeze considerabil îndeplinirea programului stabilit, ar fi cu mult mai bine dacă am avea la dispoziție un sistem „secvențial" sau „de comandă" prin care să putem reorganiza întregul ansamblu de lucrări încă neefectuate, în orice moment și în orice stare a execuției lucrărilor, și care să ofere un grafic de timp corespunzător noilor situații. Dezvoltarea unui astfel de sistem (de preferat pe calculatoare profesionale personale) va face obiectul preocupărilor noastre în etapa următoare.

Gestiunea stocurilor şi serviciile oferite clienţilor

În continuare vom menţiona o altă activitate a KITE în care SZTAKI este implicat. Deoarece fermele-membre au achiziţionat deja mii de maşini, KITE are obligaţia să asigure repararea acestora şi să înmagazineze piese de schimb. Ca urmare la Nádudvar au fost construite magazii mari, care conţin stocuri de piese pentru diferite maşini. Numărul total de tipuri de piese este de aproximativ 30 000. Procedura normală de vînzare constă în apariţia la Nádudvar a unui reprezentant al unei ferme, însoţit de un camion şi de o listă de piese dorite. Lista este prelucrată pentru a se stabili disponibilitatea pieselor căutate. Dacă codul unei piese nu este cunoscut, o anumită descriere generală este suficientă pentru ca personalul KITE să poată asigura găsirea numărului său de cod. Este întocmită apoi o a doua listă, cu piesele disponibile, cu care clientul se deplasează la magazie şi obţinute piesele. Facturarea şi plata se efectuează mai tîrziu. Dacă o anumită piesă nu este disponibilă pe loc, atunci cererea este trecută într-o listă de aşteptare, iar clienţii sînt anunţaţi deîndată ce piesele în cauză sosesc în magazie.

De asemenea KITE posedă stocuri de piese de schimb în 76 de camioane de service care circulă de la o fermă la alta de-a lungul ţării şi asigură servicii, dacă sînt solicitate de fermele membri.

Întreaga activitate, inclusiv cea administrativă adică service-ul, facturarea comenzii de rezervare, eliberări de materiale etc., impune trecerea pe calculator, aşa cum s-a făcut, încă de mulţi ani, în ţările puternic industrializate.

La prima vedere, întreaga afecere pare foarte simplă: stocurile trebuie introduse în baza de date corespunzător inventarului, iar apoi operaţii ca regăsirea, actualizarea informaţiei etc. pot fi executate prin tranzacţii de la terminale conectate on-line puse la dispoziţia utilizatorilor, în timp ce dacă sînt necesare informaţii sau trebuie efectuate operaţii asupra bazei de date în ansamblul ei, vor trebui scrise programe pentru prelucrare pe loturi. Atunci ce ne determină să fim mîndri de acest software şi să îl menţionăm? El nu asigură alocarea optimală a pieselor în magazii şi nici nu calculează rutele optime pentru camioanele de service ale KITE. Ei bine, la o privire mai atentă s-a descoperit că practica este mult mai complicată decît ne-am imaginat vreodată. Există aproximativ 90 de locuri de înmagazinare, dacă luăm în considerare şi camioanele de service (acesta fiind modul de abordare cel mai recomandabil). Unele piese sînt încă proprietatea firmelor străine, iar altele au fost deja cumpărate de KITE, procedura de facturare fiind în acest caz total diferită. Din cînd în cînd unele firme îşi modifică tehnologia, precum şi preţurile unor piese, ceea ce determină modificarea codurilor de identificare.

Piese de acelaşi tip pot avea diferite preţuri în funcţie de data la care au fost importate. Unele piese nu există în stoc la un anumit moment, dar ar putea fi înlocuite de altele. Ca urmare baza de date ar trebui să aibă structuri extensive de indicatori, care să arate cum poate fi substituită o piesă (direct cu o alta sau cu modificări minore). Dacă unele piese aparţin unui anumit proprietar, iar în stoc există altele, identice cu acestea dar aparţinînd altui proprietar, atunci este necesară implementarea unei politici referitoare la prioritatea de vînzare a pieselor.

Unele piese sînt prezente, fizic în stoc dar sînt deja reţinute prin anumite contracte, astfel încît nu sînt disponibile decît pentru parteneri bine definiţi din listele de aşteptare. Alte piese sînt prezente logic în stoc (de vreme ce sînt trecute în inventar) dar nu există fizic, deoarece nu pot fi găsite nicăieri, şi oricare ar fi cauza dispariţiei lor, baza de date trebuie încunoştiinţată asupra acestui trist eveniment, în speranţa că, într-o zi norocoasă, piesele lipsă vor reapare, şi aşa mai departe.

În continuare, este necesară, de asemenea, dezvoltarea unor algoritmi pentru comunicaţia cu furnizorul.

Este mult prea tîrziu să dai o nouă comandă atunci cînd calculatorul a găsit stocul aproape epuizat. Cererea pentru unele piese are un caracter extrem de sezonier, astfel încît ar fi inutil să cercetăm cîte astfel de piese au fost vîndute în lunile precedente. Este preferabilă studierea lunilor corespunzătoare din anii prece-

denți, dar acest lucru presupune păstrarea înregistrărilor din trecut, adică menținerea arhivelor pe mai mulți ani și combinarea datelor oferite de acestea cu alte date care indică modificarea numărului total (și a vîrstei) mașinilor cărora le aparțin piesele în cauză, astfel încît să poată fi estimat, cu o aproximație rezonabilă, numărul bucăților ce vor fi comandate.

Mai mult de 200 de programe și tranzacții au fost necesare pentru acoperirea diferitelor obiective. În plus, ar fi fost de dorit ca sistemul de echipamente și sistemul de gestiune a bazei de date pe care le-am avut la dispoziție să fie mai puternice, mai rapide și cu mai bune performanțe. Este vorba de R-11, un calculator relativ mic produs de firma VIDEOTON. El are o memorie centrală de 0,5 M bytes și 3×50 M bytes memorie externă pe discuri CDC. Sistemul de gestiune a bazei de date, denumit DMS-600, este o versiune a sistemului TRIBU dezvoltat în Franța și pus în circulație, sub licență, de VIDEOTON.

Aceasta este aproape tot ceea ce noi credem că merită să fie prezentat. Rămîne încă ceva de spus despre, să-i spunem, „filozofia" generală, a abordării noastre. Întrebarea crucială pare să fie următoarea: ceea ce facem noi în cooperare, pentru agricultura Ungariei este într-adevăr ceea ce ar trebui să facă un institut al unei Academii de științe? Oare acestea nu sînt obiective mai potrivite pentru casele de software sau chiar pentru instituțiile agricole, în timp ce noi ar trebui să „zburăm mai sus" și să ne ocupăm cu probleme mai teoretice prin care să putem construi pentru viitor? Noi credem că țara noastră este în prezent încă în faza de dezvoltare în care utilitatea automatizării și a tehnicii de calcul trebuie dovedite în practică, pe piață. Eforturile pe care le-am menționat aici urmăresc tocmai această dovadă. Apoi, mai tîrziu, pas cu pas, ne putem apropia și de lucrări teoretice și practice de un nivel științific mai înalt.

O ultimă remarcă: imaginea pe care v-am prezentat-o nu ar fi completă fără mențiunea că există și alte sisteme de producție, de mare succes, atît pentru producția de culturi agricole, cît și pentru împerecherea animalelor. Există de asemenea și alte institute de cercetare, case de software și altele ca acestea, serios implicate în automatizarea agriculturii ungare. Deși trecerea în revistă a activității acestora ar fi fost probabil instructivă, credem că este încă prea devreme să o facem. Peste 2 sau 3 ani, după ce vom mai aduna experiență, vom putea reveni din nou la această temă.

Concluzii

Deși conducătorii sistemelor de producție din agricultura ungară sînt foarte interesați, în principiu, de diferitele metode de automatizare, punctul lor de vedere este absolut practic: ei doresc să-și rezolve problemele comune, și să obțină, foarte repede, un cîștig real ca urmare a introducerii unor metode și instrumente care implică investiții costisitoare. Ca urmare modul de abordare al Institutului pentru Automatizări (SZTAKI) trebuie adaptat cu strictețe necesităților concrete, iar progresele se obțin numai treptat, muncind cu răbdare. Ar fi mai ușor și, mai profitabil din punct de vedere al rezultatelor științifice, ca investigațiile să se orienteze spre probleme mai abstracte ale agriculturii decît cele menționate, dar în acest caz SZTAKI ar pierde contactul cu necesitățile concrete, cu lumea reală. Această ultimă alternativă trebuie evitată cu orice preț.



RESURSELE INFORMAȚIONALE NAȚIONALE*

după G. Gromov
U.R.S.S.

Revista sovietică „Znanie-Sila" prezintă, în numărul său din ianuarie 1985, un fragment din cartea, în curs de apariție în editura „Nauka", a specialistului în știința calculatoarelor G. R. Gromov intitulată „Resursele informaționale: problemele exploatării industriale", în care autorul evidențiază tendințele principale pe care le manifestă dezvoltarea industriei mondiale de calculatoare și a tehnologiei informatice, implicațiile economice și sociale ale înnoirilor tehnologice pe care acestea le declanșează, caracterul lor revoluționar, concluzii pentru orientarea cercetării și a activității productive.

Ținînd seamă de interesul și actualitatea problemelor relevate de autor, publicăm textul apărut în revista „Znanie-sila".

*
* *

Eu din cunoaștere mi-am făcut meseria...
Omar Haiam
„Rubai"

Ce sînt *resursele informaționale naționale?* Ce sens concret trebuie introdus în această noțiune? Ansamblul de minereuri — minereuri metalifere, cărbunele, petrolul, gazele etc. — înmagazinate în subsolul țării, se numesc uneori *resursele minerale naționale.* Se cunosc și așa numitele *resurse reînoibile:* energia rîurilor și soarelui, masivele păduroase, terenurile agricole. Ponderea lor economică în bogăția națională a țării este clară și, de regulă, nu necesită explicații. Dar ce sens real poate fi introdus în noțiunea de „resurse informaționale"? Intuitiv noi înțelegem: resursele sînt ceea ce ne „încălzește, hrănește, îmbracă" nu imagini informaționale imateriale. „Privighetoarea nu se hrănește cu povești!" — astfel, într-o formă laconică se formula din vechime această idee, cînd se considera necesar să se oprească „fluxul de informație", pentru a se ocupa în sfîrșit de o chestiune reală — munca cu obiectele materiale.

Dar în ultimul sfert al secolului XX, *informația, sursele ei,* precum și *mijloacele de prelucrare și păstrare* a acesteia, se dovedesc pentru țările industrial dezvoltate *una dintre cele mai valoroase componente ale avuției naționale totale* sau, conform altei terminologii, o resursă națională strategic importantă.

Pentru confirmarea acestei teze se indică adesea modificarea rapidă a ponderii economice relative a Japoniei în raport cu țările capitaliste de frunte. Săracă în resurse naturale, țara insulară, în ultimul sfert de secol a depășit în dezvoltare economică pe concurenții săi incomparabil mai bogați în rezerve naturale — Anglia, R.F.G. și Franța și a ieșit pe locul doi după S.U.A. între țările capitaliste în ce privește nivelul producției industriale. Una din cauzele principale ale acestui lucru este faptul că, printre direcțiile principale de dezvoltare a economiei, în Japonia s-a produs o restructurare accelerată a mecanismului economic al țării pentru organizarea exploatării industriale, la scară largă, a resurselor informaționale naționale. Deja de un șir de ani, acolo se dezvoltă activ, la scară de stat, programul de creare către anul 2000 a așa numitei „societăți informaționale", fiecărui membru al căreia informația să-i fie la fel de accesibilă cum sînt accesibile în prezent, pe bază comercială, populației țărilor industrial dezvoltate apa și energia electrică.

Steaua denumită Pămînt

... Spuneţi, de ce s-au împrăştiat oamenii de ştiinţă pe diferite domenii şi fiecare vorbeşte într-o limbă pe care celălalt nu o înţelege? De ce am studiat totul, am descris totul şi nu cunoaştem aproape nimic?
— Scuzaţi, acesta nu este obiectul meu, eu numai colectez faptele — eu sînt statistician!
V. F. Odoevski, „Nopţi ruse".

Pe parcursul întregii istorii a dezvoltării civilizaţiei omeneşti obiectul principal al muncii l-au constituit obiectele materiale. Activitatea în afara limitelor producţiei materiale şi serviciilor, de regulă, făceau parte din categoria „cheltuielilor neproductive". Puterea economică a statului era măsurată prin resursele materiale pe care le controla acesta.

Acum 100 de ani peste 95% din numărul total al populaţiei apte pentru muncă din S.U.A. era ocupată nemijlocit în sfera producţiei materiale şi a serviciilor şi mai puţin de 5% lucra cu informaţia. Situaţia era, conform criteriilor tradiţionale, pe deplin satisfăcătoare: pentru fiecare 20 de inşi, ocupaţi cu o treabă reală — cu munca cu obiectele materiale, revenea un om care „mînuia hîrtiile". În prezent situaţia s-a modificat radical: acum deja, în medie, fiecărui om ocupat cu munca cu obiectele materiale, îi corespunde un om pentru care obiectul principal al muncii este informaţia.

Ponderea aşa numitei „sfere informaţionale" — în care intră conducătorii de toate nivelele, oamenii de ştiinţă şi specialiştii, funcţionarii din instituţii — la mijlocul anilor '80 a depăşit jumătate din totalul populaţiei apte de muncă în S.U.A. şi continuă să crească. Tendinţe analoge au loc şi în alte ţări industrial dezvoltate. Astfel, la sfîrşitul secolului XX, pentru prima oară în istoria omenirii, obiectul principal al muncii pentru majoritatea celor ocupaţi în producţia socială a ţărilor industrial dezvoltate devine informaţia. Tendinţa transferării necontenite a resurselor de muncă din sfera producţiei materiale în sfera informaţională este simptomul cel mai vizibil, dar de departe nu singurul, al schimbărilor ce se apropie şi care au căpătat deocamdată denumirea generală şi oarecum neclară de „criză informaţională".

Există oare oarecari estimări cantitative simple şi intuitive ale acestui proces social-economic complex, multilateral?

Şeful secţiei de astrofizică de la Institutul de cercetări cosmice al Academiei de Ştiinţe a U.R.S.S., membru corespondent al Academiei de Ştiinţe a U.R.S.S. I. S. Şklovski consideră că „o bună caracterizare a nivelului de dezvoltare a civilizaţiei tehnologice o poate oferi nivelul de producţie a energiei. Pentru civilizaţia terestră acest nivel va atinge în curînd 10^{20} erg./s. Dar puterea fluxului radiaţiei solare, incident pe planeta noastră, depăşeşte încă, de peste 10 mii ori, acest nivel şi constituie cca 10^{24} erg./s."

Astfel, dacă se evaluează în linii generale indicatorii energetici, în ultimii 300 de ani de creştere intensivă a producţiei şi consumului de energie, omenirea nu a depăşit încă limitele unor sutimi de procent din fondul solar incident pe planeta pămînt. Pe de altă parte, menţionează I. S. Şklovski: „Luînd în considerare numărul de emiţătoare de televiziune existente pe pămînt, puterea lor şi durata relativă a emisiunilor, se poate demonstra că Pămîntul radiază în unde metrice o putere de aproximativ 1 milion de ori mai mare decît cea pe care el ar radia-o pe cale naturală". Merită să ne gîndim la acest exemplu. În cca 2—3 decenii, datorită activităţii civilizaţiei terestre în curs de dezvoltare, o asemenea proprietate globală importantă a planetei noastre cum este puterea radiaţiei în unde radio, a crescut într-o măsură enormă. „Datorită activităţii fiinţelor inteligente, pămîntul, ca putere de radiaţie în bandă metrică, a ajuns pe primul loc printre planete, depăşind planetele gigant Jupiter şi Saturn şi rămînînd (deocamdată!) numai în urma soarelui!" — subliniază I. S. Şklovski.

Deci, fenomenul pe care îl denumim „explozie informaţională", pentru un observator din cosmosul îndepărtat ar apare ca o explozie în banda de unde metrice a unei „stele" noi, apropiată ca strălucire de Soare, în locul planetei Pămînt, rece în decurs de miliarde de ani.

Savantul american James Martin, autorul unui şir de cărţi privind tehnica de calcul, propune o altă scară pentru măsurarea ritmurilor de creştere a „sarcinii informaţionale" asupra umanităţii: „... Acum noi am atins un asemenea nivel al cunoaşterii cînd cantitatea de informaţie, care pătrund în industrie, conducere şi lumea ştiinţifică, ajunge pînă la mărimi îngrijorătoare. Presa denumeşte destul de moale şi nereuşit acest lucru „explozie informaţională", întrucît explozia se derulează rapid, în timp ce creşterea informaţiei, în perspectivă, nu are limite. Suma totală a cunoştinţelor omeneşti se modifica înainte foarte lent... în anul 1800 ea se dubla la fiecare 50 de ani, către anul 1950 se dubla la fiecare 10 ani, iar către anul 1970 se dubla la fiecare 5 ani".

Doi ani după începerea exploatării primului calculator electronic din lume, părintele ciberneticii Norbert Wiener, încerca să explice situaţia constituită către mijlocul secolului XX printr-o excursie istorică scurtă: „Ideile fiecărei epoci se reflectă prin tehnica ei. Inginerii antichităţii erau topometri, astronomi şi navigatori; inginerii secolelor XVII şi începutului secolului XVIII erau ceasornicari şi şlefuitori de lentile... Rezultatul practic, principal al acestei tehnici, bazate pe ideile lui Huygens şi Newton, a fost epoca navigaţiei, cînd pentru prima oară a fost posibil să se calculeze longitudinile* cu o precizie acceptabilă şi comerţul cu ţările de peste ocean, care anterior constituia ceva accidental şi riscant, să se transforme într-o întreprindere pusă pe baze sigure. Aceasta a fost tehnica comercianţilor. Apoi, negustorul a fost înlocuit de fabricant, iar locul cronometrului l-a ocupat maşina cu abur. De la maşina lui Newcomen aproape pînă în prezent, domeniul principal al tehnicii a fost studiul motoarelor primare... Căldura s-a transformat în energia utilă de rotaţie şi translaţie şi fizica lui Newton s-a completat cu fizica lui Rumford, Carnot şi Joule... Dacă secolul XVII şi începutul secolului XVIII este secolul ceasornicului, iar sfîrşitul secolului XVIII şi întreg secolul XIX — secolul maşinilor cu aburi, timpul actual este secolul comunicaţiilor şi conducerii. În electrotehnică există împărţirea în domenii numite într-un şir de ţări tehnica curenţilor tari şi tehnica curenţilor slabi, iar în S.U.A. şi Anglia — energetica şi tehnica telecomunicaţiilor. Aceasta şi este limita care separă secolul trecut de cel în care trăim acum".

Există motive să se presupună că deja în cursul anilor '80 cheltuielile ţărilor industrial dezvoltate pentru „tehnica curenţilor slabi" — electronica şi telecomunicaţiile — vor depăşi (acolo unde ele încă nu au depăşit) cheltuielile pentru „tehnica curenţilor tari" — energetica. Aceasta înseamnă că la începutul anilor '90 ţările industrial dezvoltate vor depăşi limita indicată de N. Wiener, care separă secolul energeticii de secolul informaţiei.

De exemplu, în cursul anilor '70, după creşterea de peste 10 ori a preţurilor de pe piaţa mondială pentru purtătorul principal de energie — petrol, cheltuielile totale pentru producerea, transportul şi consumul energiei s-au stabilizat în S.U.A., către începutul deceniului actual, la nivelul de 13% din produsul naţional brut (PNB). Pe de altă parte, cheltuielile pentru procurarea şi exploatarea tehnicii de calcul, care se estimau la sfîrşitul anilor '70 la 5% din P.N.B., vor atinge 8% către 1985 şi 13% către anul 1990.

Dacă se ia în considerare că cheltuielile pentru tehnica de calcul constituie doar cca jumătate din întregul volum al vînzărilor de echipament electronic în S.U.A., iar cheltuielile anuale pentru producerea şi exploatarea mijloacelor de telecomunicaţii erau estimate către anul 1980 la nivelul de 4—9% PNB, atunci devine evident că cheltuielile totale ale societăţii americane pentru ramurile informaţionale depăşesc vizibil cheltuielile pentru energetică.

* Pentru a estima nivelul complexităţii celei mai importante probleme de navigaţie — a determinării longitudinii în sec. XVIII este suficient să ne amintim cum Jonathan Swift ironiza în „Călătorie în Laputa..." asupra descoperirilor neverosimile pe care contemporanii lui le aşteptau de la ştiinţa viitorului: „Dar ce descoperiri mărete le-aţi primi neîndoielnic: perpetuum mobile, descoperirea medicamentului universal împotriva tuturor bolilor sau procedee de determinare a longitudinii!"

Creşterea ramurilor informaţionale ale industriei rezultă şi dintr-o comparare a volumului total al vînzărilor produselor ramurilor noi şi tradiţionale ale industriei din S.U.A., din Europa occidentală şi din Japonia în anii '80.

Astfel, se pot indica cel puţin 3 simptome diferite, fiecare dintre ele dovedind convingător începutul trecerii ţărilor industrial dezvoltate într-o etapă calitativ nouă de dezvoltare tehnică, care se obişnuieşte să se numească secolul informaţiei — în deceniul actual timpul de dublare a volumului de cunoştinţe ştiinţifice acumulate va constitui deja un an sau doi; cheltuielile materiale pentru păstrarea, transmiterea şi prelucrarea informaţiei depăşesc deja de pe acum cheltuielile analoge pentru energetică; pentru prima oară în istoria sa omenirea devine un „factor cosmic", observabil real la distanţe astronomice — nivelul radiaţiei undelor radio al planetei pămînt în unele sectoare ale benzii radio se apropie ca strălucire de nivelul radiaţiei în aceste unde emanate de către soare.

Economia informaţională

> Să admitem că este verosimil, dar este inadmisibil să se insiste asupra acestui lucru.
> Cicero „Probleme academice"

Conform datelor UNESCO, în prezent, peste jumătate din întreaga populaţie ocupată a ţărilor capitaliste cele mai dezvoltate participă direct sau indirect la procesul de producere şi diseminare a informaţiei.

Dacă productivitatea muncii muncitorilor în ramurile automatizate ale industriei prelucrătoare creştea în ultimele decenii într-un ritm de cca 80% pe deceniu, ritmurile de creştere a productivităţii muncii oamenilor ocupaţi în instituţii, aşa numiţilor „gulere albe", nu depăşeau în aceeaşi perioadă 4%.

William Ross Ashby remarca la timpul său că, în industria actuală, puterea muncitorului, care dispune în medie de 0,1 C.P., se amplifică pînă la valoarea medie de 1 000 C.P. (adică de 10 000 de ori). Gradul analog de amplificare a capacităţilor intelectuale ar da valoarea coeficientului de capacităţi intelectuale egală cu 1 milion.* La începutul anilor '60 Stafford Bear se referea la această idee a lui Ashby pentru a sublinia că „problemele care stau în faţa noastră în sfera producţiei industriale nu pot fi rezolvate doar cu un intelect neînarmat... "

Problema „înarmării intelectului" nu este simplă. După cum se ştie, majoritatea eforturilor productive ale oamenilor, ocupaţi în sectorul informaţional al producţiei sociale, are drept scop conducerea oamenilor şi maşinilor în cursul procesului de muncă. Complicarea în avalanşă a procesului de muncă şi a relaţiilor dintre participanţii la acest proces apropie societatea de pragul natural al complexităţii peste care inteligenţa, neînarmată cu mijloace pentru prelucrarea informaţiei, se dovedeşte a nu mai fi în stare de a controla eficient situaţia.

Cu ce mijloace a fost înarmată, pînă în ultimul timp, partea principală a celor ocupaţi în sfera informaţională? Telegraf, telefon, maşina de scris, iar ulterior xeroxul, iată tot ce a pătruns în sfera informaţională de pe masa revoluţiei industriale. De aceea creşterea, concomitent cu mărirea complexităţii societăţii industriale, a volumului şi a vitezei fluxurilor informaţionale care circulă în ea, a fost însoţită în esenţă prin mărirea corespunzătoare a proporţiei relative de oameni ai muncii ocupaţi în sectorul informaţional al producţiei sociale. Tocmai de aceea, oamenii muncii ocupaţi întreaga zi de muncă cu crearea şi transformarea informaţiei constituie o parte tot mai însemnată a celor ocupaţi în toate ramurile de bază ale eco-

*Se presupune că drept 100% s-a adoptat aşa numitul „coeficient de capacităţi intelectuale" al omului mijlociu; atunci amplificarea de 10 000 ori dă valoarea „1 000 000%".

nomiei. În aceste condiţii, productivitatea scăzută a muncii „gulerelor albe" apasă practic tot mai mult asupra tuturor ramurilor economiei.

O dată cu creşterea numărului relativ al celor ocupaţi în sectorul de prelucrare a informaţiei, ritmurile de creştere a eficacităţii producţiei sociale pe ansamblu în ţările industrial dezvoltate sînt în necontenită scădere.

În anul 1976 în literatura ştiinţifică s-a introdus pentru prima oară termenul „economia informaţională". Acum, influenţa crescîndă a ritmurilor de dezvoltare a industriei de prelucrare a datelor asupra indicatorilor de importanţă vitală ai economiei naţionale se constată în cele mai diferite ţări şi devine obiect al atenţiei permanente şi deosebite a oamenilor de ştiinţă, inginerilor, economiştilor, politicienilor. Necesitatea dezvoltării metodelor ştiinţifice de analiză a situaţiei devine tot mai acută. „Noi trecem acum de la societatea industrială la societatea bazată pe prelucrare informaţiei, însă, teorii economice îndreptate spre susţinerea existenţei acestei societăţi încă nu s-au creat. Pe măsură ce prelucrarea informaţiei devine o parte tot mai mare a activităţii noastre de producţie, creşte necesitatea existenţei unor metode acceptabile de determinare a valorii informaţiei şi a expresiei ei cantitative în categorii economice. Eu nu ştiu — se exprimă cu părere de rău E. Wantlend, conducătorul firmei „Tectronix", una dintre cele mai mari firme de calculatoare din S.U.A. — de unde vor apare asemenea teorii şi metode corespunzătoare şi, deocamdată, nu văd nimic în acest domeniu, care să fie, într-o măsură cît de mică, practic şi util. Dar ceva trebuie să se producă în acest sens..."

Coada şopîrlei

> Omul, servitorul şi tălmăcitorul Naturii, face şi înţelege numai atît cît poate cuprinde din ordinea Naturii; mai mult de atît el nu ştie şi nu poate nimic.
> Francis Bacon „Noul organon"

Dependenţa crescîndă a ţărilor industrial dezvoltate de sursele de informaţie — tehnică, economică, politică, militară — precum şi de nivelul de dezvoltare şi de eficienţă a utilizării mijloacelor de transmitere şi prelucrare a dus la apariţia, la începuturile anilor '80, a unei noţiuni principial noi — resursele informaţionale naţionale. Desigur, informaţia era acumulată şi preţuită întotdeauna. Nouă s-a dovedit acum creşterea vertiginoasă, constatată în ultimele decenii, în ţările industrial-dezvoltate, a importanţei economice a resurselor informaţionale pentru economia naţională, a ponderii lor în economia naţionala.

Preşedintele programului de elaborare a politicii în domeniul resurselor informaţionale, profesorul A. Ettiger de la Universitatea Harvard consideră că soseşte timpul cînd „informaţia devine o resursă la fel de importantă ca şi materialele şi energia şi, prin urmare, în raport cu această resursă trebuie formulate aceleaşi întrebări critice: cine le posedă, cine este interesat în ele, cît sînt ele de accesibile, este oare posibilă utilizarea lor comercială?"

Resursele informaţionale naţionale constituie o nouă categorie economică. Formularea corectă a problemei evaluării lor cantitative şi a legăturii lor cu alte categorii economice mai aşteaptă încă elaborări şi va necesita, probabil, eforturi comune îndelungate din partea specialiştilor şi oamenilor de ştiinţă din cele mai diferite domenii ale cunoaşterii. Deocamdată, prezintă interes examinarea, pe baza exemplului S.U.A., a acelor tendinţe care constituie reflectarea influenţei crescînde a resurselor informaţionale naţionale asupra celor mai importanţi indicatori ai dezvoltării economice a ţării.

Epuizarea rapidă a rezervelor naturale de materii prime, încă acum cîteva decenii, a pus în faţa S.U.A. problema reorientării economiei spre utilizarea mai ales a resurselor reproductibile. În anul 1970 Comisia specială pentru politica ştiinţifică, care a prezentat recomandările preşedintelui S.U.A. remarca: „resursele ştiinţifice şi tehnice ale ţării noastre constituie instrumentul cel mai puternic pentru atingerea

scopurilor sociale, politice şi economice. Repartiţia corespunzătoare a acestor resurse vitale, controlul şi conducerea lor pe termen scurt şi pe termen lung pentru rezolvarea problemelor naţionale şi internaţionale constituie o sarcină politică de maximă importanţă".

Un an după aceasta, preşedintele Academiei Naţionale de Ştiinţe a S.U.A. F. Hendler formula această idee în modul următor: „Economia noastră trebuie să se bazeze nu pe resursele naturale, ci pe inteligenţă şi pe aplicarea cunoaşterii ştiinţifice". Iar în anul 1976 această concepţie şi-a găsit reflectarea în legea S.U.A. privind politica ştiinţifică naţională, la principii organizatorice şi priorităţi.

Resursele informaţionale, ca şi cele agricole, fac parte dintr-un număr destul de limitat de resurse semnificative din punct de vedere economic, reproductibile. În cursul anilor '70 creşterea exportului agricol, ca şi creşterea cotei ramurilor cu mare consum de ştiinţă în exportul industrial a devenit manifestarea principală a reacţiei de protecţie a mecanismului economic al S.U.A. la scumpirea bruscă a materiilor prime şi resurselor energetice din import. Dacă pentru produsele tradiţionale ale industriei prelucrătoare — oţel, laminate, textile, îmbrăcăminte, încălţăminte — deficitul balanţei de comerţ exterior al S.U.A. în aceşti ani a crescut de şase ori, activul în comerţul cu produse care încorporează un mare volum de cercetare ştiinţifică — avioane, calculatoare, produse chimice — a crescut în aceeaşi perioadă de patru ori. Cel mai rapid creşte exportul de produse şi servicii ale ramurilor din industria informaţiei.

Ramurile consumatoare de ştiinţă din industrie, şi în primul rînd, industria de calculatoare, constituie obiectul unei atenţii speciale a instituţiilor conducătoare din S.U.A. În cursul ultimelor decenii ponderea S.U.A. în exportul total al ţărilor capitaliste a scăzut la o treime — de la 33% în 1948 la 11% în 1980. „Către anul 1977, explică economistul sovietic E. V. Kiricenko, R.F.G. a depăşit, iar Japonia s-a apropiat nemijlocit de S.U.A. în ce priveşte valoarea exportului maşinilor şi utilajelor în ansamblu, S.U.A. a ocupat doar locul IV printre ţările capitaliste în ce priveşte volumul valoric al exportului de maşini unelte şi utilaje pentru forjare şi presare... locul al III-lea privind exportul de utilaje energetice pentru centrale electrice...". Cu alte cuvinte, S.U.A. cedează treptat celor mai apropiaţi concurenţi aproape toate sectoarele tradiţionale ale pieţii capitaliste mondiale. Toate, în afară de cel principal — sectorul tehnologiei informaţionale.

Cedînd urmăritorilor săi „coada-şopîrlei" — întîietatea într-un evantai întreg de ramuri tradiţionale ale industriei, S.U.A. s-au desprins de ei aproape cu un ordin de mărime în ce priveşte indicatorii principali în industria informaţională. Conform datelor publicate de Institutul pentru S.U.A. şi Canada al Academiei de Ştiinţe a U.R.S.S., cu toată presiunea crescîndă din partea industriei de calculatoare a Japoniei şi R.F.G., S.U.A. tot mai controlau la începutul anilor '80 cca 80% din piaţa de calculatoare a ţărilor capitaliste; sub raport valoric, jumătate din calculatoarele instalate în lumea capitalistă se află în S.U.A., iar pe 70% din cealaltă jumătate este ştampila „Made in U.S.A.". Centrul luptelor economice pentru întîietate pe piaţa capitalistă mondială se deplasează necontenit spre domeniul mijloacelor de prelucrare a informaţiei. Monopolurile din S.U.A. au reuşit să cucerească şi să menţină un timp îndelungat poziţiile de comandă în acest domeniu, însă acum această situaţie se dovedeşte complet inacceptabilă pentru corporaţiile informaţionale naţionale din Europa occidentală şi Japonia a căror putere este în creştere. Conducătorii acestor ţări întreprind cele mai energice, iar uneori şi cele mai extravagante, eforturi pentru a activiza procesul de producţie a mijloacelor naţionale de prelucrare electronică a datelor, precum şi introducerea lor în masă în mecanismul economic al ţării. După cum remarca în octombrie 1980 redactorul revistei „Electronique", „La Paris rareori se întîmplă o zi în cursul căreia un înalt funcţionar sau altul sau un reprezentant al guvernului, inclusiv preşedintele ţării, să nu amintească cetăţenilor că preţul dispozitivelor de memorie pe unitatea de informaţie stocată scade necontenit. Toate aceste persoane oficiale sînt convinse că viitorul Franţei depinde în mare măsură de faptul dacă ea va şti să devină în următorii cinci ani un factor important în producţia mondială de circuite integrate, calculatoare, mijloace de telecomunicaţii şi echipamente automate pentru instituţii". În ultimul timp a suscitat deosebit de multe discuţii provocarea lansată S.U.A. din Ţara Soarelui Răsare — programul de creare în Japonia a calculatoarelor din generaţia a 5-a.

În afară de influența evidentă asupra capacității de concurentă a ramurilor mari consumatoare de inteligență ale industriei, resursele informaționale au o acțiune hotărîtoare și asupra unui asemenea articol, important sub aspect economic și politic al comerțului exterior, cum este balanța de brevete și licențe. Așa numitul „export invizibil" servește drept unul din indicatorii determinanți ai nivelului curent al „desprinderii tehnologice" a S.U.A. de cei mai apropiați concurenți de pe piața capitalistă. Iată părerea lui G. Pollac care era pe atunci conducătorul secției de probleme tehnico-științifice din departamentul de stat, părere caracteristică pentru S.U.A. anilor '60: „Înțelegerea împrejurării că viabilitatea economiei naționale depinde acum, în măsură considerabilă, de calitatea și gradul de utilizare de către ea a științei și tehnicii a adus pe prim plan compararea potențialelor tehnologice ale statelor și, respectiv, problema „desprinderii tehnologice". Această desprindere are astăzi pentru diplomat aceeași importanță pe care o avea acum cîteva generații compararea dimensiunilor armatelor".

Estimările oficiale cunoscute ale balanței de brevete și licențe, măsurate în miliarde dolari, reflectă doar o mică parte din fluxul total de soluții tehnico-științifice și documentații tehnologice care circulă pe piața industrială mondială. O mare parte dintre acestea nu sînt prinse în statistica oficială, deoarece circulă prin canalele interne ale corporațiilor transnaționale (CTN). Cînd, în anul 1982, o comisie specială pentru legături științifice și securitate națională a Academiei Naționale a S.U.A. a încercat să examineze posibilitatea controlării fluxurilor informaționale din interiorul firmelor, acest fapt a fost întîmpinat de conducătorii CTN cu o iritare fățișă. L. Branscomb. vicepreședintele pentru munca științifică a celei mai mari firme de calculatoare din lumea capitalistă, IBM, a declarat că orice control al transmiterii secretelor tehnico-științifice, păstrate cu minuțiozitate, între întreprinderile acestui concern, aflate în S.U.A. și în alte țări, poate avea pentru el urmări catastrofale, deoarece cca jumătate din profituri și venituri sînt obținute de IBM de la filialele din străinătate: „Oricît ar fi de paradoxal, scrie Branscomb, specialistul care a ocupat primul loc printre angajații concernului nostru, ca număr și importanță a invențiilor, este un inginer din laboratorul nostru din R.F.G.".

Acest incident birocratic de mică importanță ca atare, radiografiază totuși particularitățile caracteristice ale etapei fin voalate a „colonialismului informațional", care înlocuiește formele mai timpurii și exterior mai vizibile, ale neocolonialismului industrial și care se manifestă prin aceea că în afara metropolelor sînt amplasate producțiile cu consum mare de materiale și energie și întreprinderile industriale cu tehnologii ecologic nocive, iar spre metropolă se îndreaptă în schimb fluxul specialiștilor locali cei mai talentați („scurgerea sau exodul creierelor").

Astăzi, toate acestea nu mai sînt suficiente. Se desfășoară o luptă silențioasă, dar îndîrjită, între CTN pentru controlul asupra celor mai valoroase resurse, din cele cunoscute pînă în prezent — resursele informaționale naționale. „Noi ne avîntăm spre alte țări, explică unul din conducătorii CTN americane „Cincinnati Milocron" — una dintre cele mai mari firme constructoare de mașini-unelte din S.U.A., care controlează cca 40% din piața americană de roboți industriali — nu pentru a folosi avantajele cheltuielilor mai scăzute. Noi ne introducem acolo, deoarece există rezerve intelectuale pe care trebuie să le captăm, pentru a avea posibilitatea să concurăm cu alte CTN."

Tendința pentru ararea activă a „ogorului informațional" străin a dus deja la o situație paradoxală, în aparență: cheltuielile monopolurilor americane pentru lucrările de cercetări științifice și de experimentare-proiectare (LCSEP) în străinătate cresc mai rapid decît aceleași cheltuileli pe ansamblul industriei S.U.A.... Personalul local constituie 92% din cei ocupați în cercetările din alte țări, iar specialiștii care sînt trimiși de firmele mamă sînt mai ales lucrători de administrație.

Cca 65% din toate cheltuileile CTN americane pentru LCSEP efectuate în afara țării revin filialelor din R.F.G., Anglia și Canada. Ironia amară a soartei constă, în cazul de față, probabil, în aceea că tot mai frecvent, într-o țară sau alta, care de secole pompa resursele naturale din coloniile împrăștiate în lumea întreagă, apare un manager cu educație rafinată de la Harvard* și, după cele mai noi canoane ale

* Cunoscuta Universitate americană.

organizării științifice a muncii, colectează și întreține în condițiile unui confort maxim realizabil, într-un „corral" din beton armat, elita înalt productivă a intelectualilor locali. Anual, cînd are loc „sezonul de strîngere a recoltei" de astă dată nu condimente, seva heveiei, petrol, trestie de zahăr sau bumbac, ci un produs incomparabil mai valoros — rezultatul exploatării resurselor informaționale naționale, conservat în brevete și în referatele LCSEP — se expediază, în containere de piele imprimate cu aur, peste Ocean.

Mărirea în continuare a complexității mecanismului economic al țărilor industrial dezvoltate a ridicat la rangul celor mai importanți factori ai dezvoltării economice nivelul culturii organizatorice și calitatea conducerii activității profesionale a oamenilor. După cum se subliniază în cercetările Institutului de Economie Mondială și Relații Internaționale al Academiei de Științe U.R.S.S., în ultimul timp la termenul „discrepanță tehnologică" s-a adăugat cel de „discrepanță de conducere", a apărut problema eficacității comparative a conducerii.

Arta, metodele și tehnologia rezolvării practice a problemelor de conducere este reunită în noțiunea de management. Acum 30 de ani în S.U.A. au început să se efectueze primele cercetări comandate privind introducerea metodelor de conducere cu calculatoare. Inițial, aceste lucrări erau efectuate numai de liderul domeniului — firma „Arthur D. Little", însă, deja la începutul anilor '70, în S.U.A. au apărut sute de organizații, ocupate independent de elaborarea metodologiilor noi de cercetare a problemelor și metodelor de adoptare a deciziilor de conducere. În total, conform datelor Institutului pentru S.U.A. și Canada al Academiei de științe a U.R.S.S., în S.U.A. funcționează 2—3 000 firme consultative. Numărul total al celor ocupați în această ramură a economiei S.U.A., legată nemijlocit de exploatarea industrială a resurselor informaționale naționale, a depășit o jumătate de milion; volumul vînzărilor de servicii consultative se evaluează la miliarde dolari și crește într-un ritm de 15% anual. Totodată, un sfert din volumul total al lucrărilor se realizează în baza contractelor cu clienți de peste hotare — firme particulare și organizații guvernamentale.

Astfel, pe piața industrială mondială rezultatele exploatării industriale a resurselor informaționale naționale sînt prezentate în prezent prin trei genuri principale de export: exportul rezultatelor LCSEP materializate în produsele cu conținut mare de inteligență ale industriei; așa numitul „export invizibil" al rezultatelor LCSEP — brevete, licențe, etc. și exportul managementului — vînzarea tehnologiei în domeniul organizării și conducerii producției.

Această ramură a economiei naționale se lărgește vertiginos, iar împreună cu ea crește la fel de impetuos, — în raport cu toate celelalte resurse naționale — valoarea resurselor informaționale, a acestui factor puternic, nou, al economiei actuale.

*
* *

Calculatorul se transformă treptat dintr-un monstru misterios, cu care oamenii puteau să comunice numai prin reprezentanții unei caste profesionale înguste de preoți, care se numeau programatori, într-un instrument individual simplu și ușor de înțeles, accesibil exploatării cotidiene în viață, la muncă, la învățătură — în toate domeniile, fără excepție, ale activității umane. Și, prin aceasta, marchează noua etapă de dezvoltare a societății.

MINICALCULATOARELE CORAL ŞI INDEPENDENT

Procesorul central. Ghid pentru utilizatori, depanatori

ing. T. Geber
(coordonare ciclu)
ing. E. Miclescu
IIRUC

CUPRINS

2.5. Instrucţiuni diverse

2.5.1. Instrucţiuni de comunicare între modul curent şi cel anterior

Instrucţiunile MFPI, MFPD, MTPI şi MTPD au sens doar pentru minicalculatoarele dotate cu management de memorie şi vor fi descrise în capitolul corespunzător 3.2 (paragraful 3.2.4).

2.5.2. Instrucţiuni de poziţionare a indicatorilor de condiţii

Instrucţiunile din această categorie permit ştergerea sau setarea unuia sau a mai multor indicatori de condiţii (N, Z, V şi C din PSW, poziţiile 0—3). Forma generală a instrucţiunilor este următoarea:

000240+X

unde X este o cantitate binară pe 5 biţi (biţii 0—4 ai instrucţiunii) care codifică operaţia de executat (ce indicatori sînt afectaţi şi în ce constă modificarea lor).

	4	3	2	1	0
X:	0/1	N	Z	V	C

Biţii 0, 1, 2 şi 3 arată dacă indicatorul C, V, Z şi respectiv N este afectat (se observă că ordinea lor corespunde cu cea din PSW):

0 ⇒ indicatorul respectiv nu este afectat;
1 ⇒ indicatorul respectiv este afectat.

Bitul 4 arată modul în care sînt modificaţi indicatorii specificaţi în biţii 0—3:

0 ⇒ indicatorii sînt şterşi (forţaţi în 0);
1 ⇒ indicatorii sînt setaţi (forţaţi în 1).

---

* Ciclu coordonat de ing. TOMA GEBER (IIRUC)
** Continuă din AMC 48, pag. 173—241.

Scriindu-l corespunzător pe X se poate șterge sau seta orice combinație de indicatori. Totuși nu au mnemonici decît cazurile mai des întîlnite și utilizate:

MNEMINICĂ ȘI NUME	X	COD
CLC — clear C (ștergere C)	01	000241
CLV — clear V (ștergere V)	02	000242
CLZ — clear Z (ștergere Z)	04	000244
CLN — clear N (ștergere N)	10	000250
CCC — clear all condition code bits (ștergere toți indicatorii de condiții)	17	000257
SEC — set C (setare C)	21	000261
SEV — set V (setare V)	22	000262
SEZ — set Z (setare Z)	24	000264
SEN — set N (setare N)	30	000270
SCC — set all condition code bits (setare toți indicatorii de condiții)	37	000277
NOP — no operation (nici o operație)	00	000240

În cazul particular XX=00 se obține o *instrucțiune inefectivă* (NOP, cu codul 000240).

2.5.3. Alte instrucțiuni

2.5.3.1. Instrucțiunea HALT

mnemonică și cod: | HALT 000000 |

— *nume:* halt (oprire).

— *operație:* este oprită derularea programului executat de unitatea centrală și controlul procesorului este acordat consolei (panoului de comandă). La panou se afișează adresa imediat următoare instrucțiunii HALT și conținutul registrului general RO. PC conține adresa instrucțiunii care urmează după HALT. Reluarea programului este posibilă numai cu start de la panoul de comandă (restart, continue).

Instrucțiunea HALT este executabilă numai în modul de lucru KERNEL; în USER, instrucțiunea e tratată ca instrucțiune ilegală (va provoca o întrerupere internă cu vectorul 4).

— *poziționare indicatori de condiții:* neafectați.

Instrucțiunea HALT este utilizată în programele de test pentru a marca erori majore, ce nu pot fi controlate de program. De asemenea, instrucțiunea poate fi folosită ori de cîte ori se dorește oprirea programului într-un anumit punct (scriind un cuvînt cu 0 — codul instrucțiunii HALT — după instrucțiunea la care se dorește oprirea programului).

2.5.3.2. Instrucțiunea WAIT

— *mnemonică și cod:* | WAIT 000001 |

— *nume:* wait for interrupt (așteptare întrerupere).

— *operație:* procesorul intră în așteptare, renunțînd la utilizarea busului (busul este acordat exclusiv DMA-urilor, permițîndu-se o viteză mai mare de transfer între aceste dispozitive și memoria centrală). Instrucțiunea este utilizată atunci cînd se

așteaptă o întrerupere externă (de la un periferic). PC conține adresa următoarei instrucțiuni, aceasta fiind salvată pe stivă atunci cînd este declanșată secvența de întrerupere (v. paragraful 2.6); revenirea din rutina de tratare a întreruperii (cu o instrucțiune RTI) va determina reluarea programului cu instrucțiunea care urmează instrucțiunii WAIT.

— *poziționare indicatori de condiții:* neafectați.

2.5.3.3. Instrucțiunea RESET

— *mnemonică și cod:*

RESET	000005

— *nume:* reset (inițializare)

— *operație:* determină activarea semnalului INIT pe bus un anumit timp, permițînd inițializarea tuturor dispozitivelor periferice și a managementului de memorie. În modul de lucru User instrucțiunea este inefectivă.

— *modificare indicatori de condiții:* neafectați.

2.5.3.4. Instrucțiunea SPL

— *mnemonică și cod:*

SPL	00023N

— *nume:* set priority level (modificare nivel de prioritate)

— *operație:* PSW 7—5 ← N

Cei trei biți mai puțin semnificativi ai instrucțiunii (reprezentînd noul nivel de prioritate N) sînt încărcați în pozițiile corespunzătoare din registrul de stare program PSW (v. 2.1.1.); se poate forța oricare dintre cele 8 nivele de prioritate (numerotate de la 0 la 7). Instrucțiunea este efectivă numai în modul de lucru Kernel. Reprezentarea sintactică:

SPL N.

— *modificare indicatori de condiții:* neafectați.

Observație: această instrucțiune nu este emulată la minicalculatoarele INDEPENDENT și CORAL. Ea face totuși parte din setul complet de instrucțiuni PDP-11.

2.6. Întreruperi

2.6.1. Declanșarea unei întreruperi și efectul ei

Prin *întrerupere* se înțelege suprimarea temporară sau definitivă a execuției unui program în urma apariției unui eveniment așteptat sau neașteptat și saltul automat la un program special de tratare, cu posibilitatea revenirii la programul inițial. Rezultă că, după apariția unui semnal de întrerupere, procesorul va trebui să execute următoarele operațiuni:

— să salveze starea programului întrerupt pentru a se face posibilă revenirea după tratarea întreruperii; în acest scop se salvează registrul de stare program (PSW) și registrul de adresă program (PC) pe stivă, conservîndu-se astfel vechea stare a programului întrerupt;

— să preia o *nouă stare program* corespunzătoare execuției programului special de tratare a întreruperii.

Întreruperile se clasifică astfel:

a) *întrerupere externă,* de la o unitate periferică cuplată la busul calculatorului;

b) *întrerupere internă*, provocată de:

b1) o instrucţiune specială;

b2) apariţia unei erori:

— eroare de dialog pe bus (TIME-OUT);

— eroare paritate memorie;

— eroare detectată de managementul de memorie;

— căderea tensiunii de alimentare;

— eroare de adresare impară;

— instrucţiune ilegală (de exemplu JMP sau JSR cu modul 0 de adresare a destinaţiei, încercarea de a executa HALT în USER);

— instrucţiune rezervată (detecţia unui cod neatribuit niciunei instrucţiuni de repertoriu);

— eroare depăşire în virgulă flotantă sau întrerupere de la procesorul de virgulă mobilă;

— depăşire stivă în KERNEL (STACK OVERFLOW).

Fiecărui tip de întrerupere îi corespunde un *VECTOR*. Vectorul este o adresă logică de la începutul memoriei centrale de la care procesorul urmează să preia noua stare program corespunzătoare rutinei de tratare a întreruperii respective. Avînd în vedere că de la adresa dată de vector trebuiesc luate 2 cuvinte (un nou PSW şi un nou PC), vectorii vor fi multipli de 4. Prima locaţie de la adresa dată de vector va conţine noul PC, iar locaţia următoare — noul PSW.

Deci, efectul declanşării unei întreruperi este următorul:

↓(SP) ←PSW
↓(SP) ←PC
} salvarea stării program întrerupte pe stivă

PC ←(vector)
PSW (vector+2)
} preluarea noii stări program, cu observaţia că în PSW 12—13 se copiază PSW 14—15 ai stării program salvate (deci modul de lucru anterior devine modul curent care a fost întrerupt).

Aceste operaţii sînt executate automat de microprogramul de emulare cu ajutorul unei microrutine speciale, apelată ori de cîte ori apare o întrerupere. Rezultatul este acelaşi ca şi cum s-ar excuta secvenţa de program:

```
MOV PS, —(SP)
MOV PC, —(SP)
MOV @ # (VECTOR+2), PS
JMP @ # VECTOR.
```

Următoarea instrucţiune executată va fi cea adresată de cantitatea conţinută la adresa *VECTOR*, aceasta trebuind să fie prima instrucţiune a programului de tratare a întreruperii.

Pentru a lucra cu întreruperi trebuiesc efectuate anterior următoarele operaţii pregătitoare:

a) încărcarea registrului SP cu adresa vîrfului de stivă (iniţializarea stivei);

b) pregătirea zonei de memorie de la adresa dată de vectorul corespunzător întreruperii vizate:

```
VECTOR   : NOU PC
VECTOR+2 : NOU PSW
```

În anexa 2 este prezentată lista vectorilor corespunzători principalelor întreruperi (interne sau externe) care pot apărea în timpul funcţionării sistemului.

Observaţie: dacă este activată procedura de relocare a adreselor (MANAGEMENT ON, vezi capitolul 3.2.), atunci şi vectorii de întrerupere sînt relocaţi; acest lucru asigură vectori de întrerupere separaţi pentru partiţiile sistem şi utilizatori.

2.6.2. INSTRUCȚIUNI DE ÎNTRERUPERE

Există 4 instrucțiuni speciale cu ajutorul cărora utilizatorii pot întrerupe derularea programului lor în diferite scopuri. Toate instrucțiunile au același efect (descris în paragraful anterior), ele diferind prin vectorii corespunzători (deci prin adresele de la care se încarcă noua stare program).

2.6.2.1. Instrucțiunea BPT

— *mnemonică și cod:*

BPT	000003

— *nume:* breackpoint trap.
— *vector:* 14
— *utilizare:* instrucțiunea este folosită în special pentru a apela rutine de depanare a programelor.

2.6.2.2. Instrucțiunea IOT

— *mnemonică și cod:*

IOT	000004

— *nume:* input/output trap.
— *vector:* 20
— *utilizare:* instrucțiunea este în special utilizată pentru apelarea rutinelor ce controlează transferurile cu echipamentele periferice (impropriu se poate spune că această instrucțiune joacă rol de instrucțiune de intrare/ieșire sub sistemul de operare).

2.6.2.3. Instrucțiunea EMT

— *mnemonică și cod:*

EMT	104000÷104377

— *nume:* emulator trap.
— *vector:* 30.
— *utilizare:* cu ajutorul acestei instrucțiuni se poate realiza un salt la o rutină a cărei adresă de început se află la adresa 30; byte-ul inferior al acestei instrucțiuni poate lua orice valoare (între 000 și 377) ce poate fi utilizată pentru a transmite informații speciale rutinei apelate. Instrucțiunea EMT este utilizată frecvent de sistemul software și de aceea nu e recomandată folosirea ei de utilizatori.

2.6.2.4. Instrucțiunea TRAP

— *mnemonică și cod:*

TRAP	104400 ÷ 104777

— *nume:* trap.
— *vector:* 34
— *utilizare:* instrucțiunea este identică cu EMT, diferind doar prin cod și vectorul asociat. Observațiile de la EMT rămîn valabile.

Observație: pentru toate instrucțiunile de întrerupere, indicatorii de condiții N, Z, V și C sînt încărcați cu valorile corespunzătoare de la adresa VECTOR+2 (deci la preluarea noului PSW).

2.6.3. INSTRUCȚIUNI DE ÎNTOARCERE DIN ÎNTRERUPERE

Instrucțiunile de întoarcere din întrerupere se folosesc la sfîrșitul rutinei de tratare a unei întreruperi pentru revenirea în programul întrerupt exact în punctul descris de starea salvată în momentul declanșării secvenței de întrerupere; deci efectul lor este următorul:

$$PC \leftarrow (SP)\uparrow$$

$$PC \leftarrow (SP)\uparrow$$

Rezultatul este același ca și cum s-ar fi executat secvența de instrucțiuni:

```
MOV (SP)+, PC
MOV (SP)+, PS
```

Există două instrucțiuni de întoarcere din întrerupere al căror efect este cel descris mai sus.

2.6.3.1. Instrucțiunea RTI

— *mnemonică și cod:* | RTI 000002 |

— *nume:* return from interrupt (întoarcere din întrerupere).

2.6.3.2. Instrucțiunea RTT

— *mnemonică și cod:* | RTT 000006 |

— *nume:* return from interrupt (întoarcere din întrerupere).

Observații: 1) instrucțiunile RTI și RTT diferă numai din punctul de vedere al interpelării bitului T din registrul de stare program (PSW 4) (vezi 2.6.4);

2) în modul de lucru User, RTI și RTT nu modifică PSW 5—7 (NIT), PSW 12—13 (modul anterior) și PSW 14—15 (modul curent) (vezi 2.6.5);

3) indicatorii de condiții sînt încărcați cu valorile corespunzătoare găsite pe stivă.

2.6.4. INTERPRETAREA BITULUI TRAP (T)

Dacă bitul T din registrul de stare program este setat (PSW4=1), procesorul va executa o întrerupere de tipul BTT, cu vectorul 14. Dacă, la reîncărcarea PSW-ului, T este setat din nou, se va executa o nouă întrerupere, tot cu vectorul 14. Bitul T nu poate fi poziționat decît prin modificarea conținutului PSW în urma unei întreruperi sau întoarcere din întrerupere (RTI sau RTT); T nu poate fi modificat prin adresarea directă a PSW-ului (MTPS). De asemenea, bitul T poate fi setat (forțîndu-se astfel o ulterioară întrerupere) de microprogramul de emulare la întîlnirea unei condiții speciale (de exemplu: depășirea stivei la INDEPENDENT-100).

Deosebirea dintre instrucțiunile RTI și RTT constă în modul de interpretare a bitului T:

— dacă în urma unei instrucțiuni RTI bitul T (trap) este setat, se va executa imediat o întrerupere cu vectorul 14;

— dacă în urma unei instrucțiuni RTT bitul T este setat, întreruperea cu vectorul 14 nu se execută imediat ci după ce se execută instrucțiunea adresată de noul PC; aceasta are aplicație în depanarea programelor.

2.6.5. RESTRICȚIILE MODIFICĂRII PSW-ULUI

Registrul de stare program poate fi modificat pe 3 căi:
1) prin adresare directă (MTPS sau cu adresa 777776);
2) prin încărcare de la vector în urma unei întreruperi;
3) prin încărcare de pe stivă la revenirea din întrerupere.

Avînd în vedere influența directă a unor indicatori din PSW asupra funcționării procesorului s-au pus anumite restricții în ceea ce privește modificarea lor.

Tabelul din fig. 2-7 prezintă modul în care este permisă modificarea diferitelor cîmpuri din PSW.

POZIȚIILE DIN P S W	POSIBILITATE MODIFICARE PRIN ADRESARE DIRECTĂ	MODIFICARE PRIN ÎNTRERUPERE	MODIFICARE PRIN ÎNTOARCERE DIN ÎNTRERUPERE
PSW 0—3 (INDICATORI DE CONDIȚII: N, Z, V, C)	DA	ÎNCĂRCAȚI DE LA ADRESA DATĂ DE VECTORUL DE ÎNTRERUPERE (VECTOR +2)	ÎNCĂRCAȚI DE PE STIVĂ
PSW 4 (TRAP: T)	NU	ÎNCĂRCAT DE LA ADRESA DATĂ DE VECTORUL DE ÎNTRERUPERE (VECTOR +2)	ÎNCĂRCAT DE PE STIVĂ
PSW 5—7 (NIVELUL DE ÎNTRERUPERE: NIT)	DA ÎN KERNEL NU ÎN USER	ÎNCĂRCAȚI DE LA ADRESA DATĂ DE VECTORUL DE ÎNTRERUPERE (VECTOR +2)	ÎNCĂRCAT DE PE STIVĂ NUMAI ÎN KERNEL; ÎN USER NEMODIFICAȚI
PSW 12—13 (MODUL ANTERIOR: PM)	NU	COPIAȚI DIN PSW 14—15 ACTUAL	ÎNCĂRCAȚI DE PE STIVĂ NUMAI ÎN KERNEL; ÎN USER NEMODIFICAȚI
PSW 14—15 (MODUL CURENT: CM)	NU	ÎNCĂRCAȚI DE LA ADRESA DATĂ DE VECTORUL DE ÎNTRERUPERE (VECTOR +2)	ÎNCĂRCAȚI DE PE STIVĂ NUMAI ÎN KERNEL; ÎN USER NEMODIFICAȚI

Fig. 2-7. Restricțiile modificării PSW

2.6.6. EXEMPLE

Exemplele următoare vor viza numai întreruperile interne.

a) *Declanşarea unei întreruperi cu ajutorul instrucţiunii IOT şi revenirea corespunzătoare:*

— se depune în memorie: noua stare program (la adresa dată de vector: 20), programul ce va fi întrerupt (la adresa 1000) şi programul de tratare a întreruperii şi revenire (la adresa 2000).

```
  20  002000                   (noul PC)
  22  000000                   (noul PSW)
1000  012706 001000            MOV #1000, SP (iniţializare stivă)
1004  000004                   A1 IOT        (întrerupere)
1006  000776                   BR A1
2000  000002                   RTI           (revenire)
```

— se lansează programul de la adresa 1000.

În cazul funcţionării corecte programul va trebui să se bucleze între adresele 1000 (programul principal, întrerupt) şi 2000 (programul de tratare a întreruperii); pe pas cu pas (instrucţiune cu instrucţiune) adresele program vor trebui să evolueze în secvenţa:

1000, 1004, 2000, 1006,
1004, 2000, 1006,
1004, 2000, 1006, etc.

Pointerul stivei (SP) va bate pasul pe loc, salvarea vechiului PSW făcîndu-se la adresa 776, iar salvarea adresei de revenire la 774 (adresa de revenire fiind cea de după IOT: 1006).

b) *simularea unei erori*

În cazul în care apare o eroare UC (BUSERR), procesorul execută o întrerupere cu vectorul 4. Cele mai simple moduri de a simula o eroare sînt:

— adresă impară (instrucţiune pe cuvînt cu adresă de operand impară);

— referirea unei adrese de serviciu inexistentă (de exemplu 160 000).

Programul arată identic ca la punctul a), doar că vectorul întreruperii e 4, iar instrucţiunea IOT trebuie înlocuită cu o instrucţiune care să simuleze eroarea:

```
   4  002000
   6  000000
1000  012706 001000            MOV #1000, SP
1004  020037 003001            CMP RO, @ # 3001 (adresare impară)
1010  000775                   BR   Al
2000  000002                   RTI
```

Pentru adresă inexistentă (TIME-OUT), instrucţiunea de la adresa 1004 se înlocuieşte cu:

```
1004  010037 160000            A1 MOV R0, @ # 160000
```

Programul se buclează, RTI realizînd întoarcerea în programul principal la adresa 1010.

c) *Verificarea instrucţiunii RTT cu bitul T (PSW 4) setat:*

— programul este următorul (vectorul de întrerupere pentru TRAP fiind 14):

```
   14  010000
   16  000000
 1000  012700 070707              MOV #70707, R0
 1004  005001                     CLR R1
 1006  012706 001000              MOV #1000, SP
 1012  012746 000020              MOV #20, -(SP)
 1016  012746 002000              MOV #2000, -(SP)
 1022  000006                     RTT
 2000  005201                A2   INC  R1
 2002  001001                     BNE  A1
 2004  005100                     COM  R0
 2006  000774                A1   BR     A2
10000  010037  xxxxxx             MOV R0,    # PANOU
10004  000006                     RTT
```

— programul se lansează de la adresa 1000 și, în caz de funcționare normală, va alterna la panou informația 070707 și complementul ei (107070) cu o anumită temporizare;

— programul are următoarele părți componente:

1) partea de pregătire (adresa 1000) în care se execută operațiile: se încarcă R0 cu informația de afișat la panou; se inițializează R1 (contor pentru temporizare); se inițializează pointerul stivei; se pregătește stiva (adresele 1012 și 1016) pentru a poziționa bitul T prin simularea unei reîntoarceri din întrerupere cu RTT; RTT va executa un salt la adresa 2000;

2) programul de temporizare (adresa 2000): temporizarea este creată prin incrementarea lui R1 pînă ce acesta ajunge iar în 0; dacă (R1)=0 saltul BNE nu se mai execută și cantitatea de afișat (R0) este complementată; datorită bitului T din PSW poziționat anterior, după fiecare instrucțiune din această secvență de program se va executa o întrerupere cu vectorul 14, (deci la programul de tratare de la adresa 10000);

3) partea de afișare și tratare a întreruperii (adresa 10000): se execută afișarea conținutului lui R0 la panoul de comandă al calculatorului; xxxxxx este adresa unui registru de LED-uri de la panou:

$$\text{xxxxxx} = \begin{cases} 170\,002 \text{ pentru INDEPENDENT-100} \\ 177\,570 \text{ pentru CORAL} \end{cases}$$

Revenirea din întrerupere cu RTT va repoziționa bitul T și, după execuția unei noi instrucțiuni din secțiunea 2 a programului se va declanșa o nouă întrerupere (v. 2.6.4.). Rezultă că se vor afișa alternativ la panou informațiile 070707 și 107070, cu o temporizare de aproximativ 4—5 secunde (rulat pe INDEPENDENT-100).

2.7. Instrucțiuni în virgulă flotantă

2.7.1. DEFINIREA OPERANZILOR

Instrucțiunile prezentate în continuare operează cu numere flotante în simplă precizie (32 biți), deci fiecare operand va ocupa cîte 2 cuvinte de memorie. Modul de reprezentare a numerelor în virgulă flotantă a fost arătat, pe larg, în paragraful 1.3.2.2.

Operanzii (duble cuvinte) se pot găsi numai în memoria centrală a calculatorului și anume într-o ministivă definită de conținutul registrului general R utilizat de instrucțiune:

— la adresele (R) și (R)+2 se află argumentul B, adică al doilea operand din punctul de vedere al operației aritmetice (scăzătorul pentru scădere sau împărțitorul pentru împărțire); la adresa (R) se află ponderile superioare ale argumentului;

— la adresele (R)+4 și (R)+6 se află argumentul A, adică primul operand din punctul de vedere al operației aritmetice (descăzutul, respectiv deîmpărțitul); la adresa (R)+4 se află ponderile superioare ale argumentului.

Rezultatul operației se depune în locul argumentului A, adică la adresele (R)+4 (ponderile superioare) și (R)+6 (ponderile inferioare). Conținutul registrului genaral R este incrementat cu 4, astfel ca la sfîrșitul operației să indice adresa rezultatului.

Instrucțiunile sînt întreruptibile, execuția lor fiind reluată în întregime la revenirea din întrerupere.

Aceste instrucțiuni în virgulă flotantă pot fi utilizate pe unele minicalculatoare care nu au un procesor specializat în tratarea numerelor flotante (de exemplu INDEPENDENT-100 și CORAL 4011 A); ele sînt microprogramate și corespund celor 4 operații aritmetice de bază.

2.7.2. OPERAŢIILE ÎN VIRGULĂ FLOTANTĂ

2.7.2.1. Instrucţiunea FADD

— *mnemonică şi cod:* | FADD 07500R |

— *nume:* floating add (adunare în virgulă flotantă).
— *operaţie:* **A←A+B** şi **R←(R)+4**

Argumentul A este adunat cu argumentul B, rezultatul depunîndu-se în locul argumentului A, deci:

[(R)+4, (R)+6]←[(R)+4, (R)+6]+[(R), (R)+2].

2.7.2.2. Instrucţiunea FSUB

— *mnemonică şi cod:* | FSUB 07501R |

— *nume:* floating substract (scădere în virgulă flotantă).
— *operaţie:* **A←A—B** şi **R←(R)+4**

Din argumentul A se scade argumentul B, rezultatul depunîndu-se în locul argumentului A, deci:

[(R)+4, (R)+6]←[(R)+4, (R)+6]−[(R), (R)+2].

2.7.2.3. Instrucţiunea FMUL

M V | FMUL 07502R |

— *nume:* floating multiply (înmulţire în virgulă flotantă)
— *operaţie:* **A←A×B** şi **R←(R)+4**

Se înmulţesc argumentele A şi B, rezultatul depunîndu-se în locul argumentului A, deci:

[(R)+4, (R)+6]←[(R)+4, (R)+6] × [(R), (R)+2].

2.7.2.4. Instrucţiunea FDIV

— *mnemonică şi cod:* | FDIV 07503R |

— *nume:* floating divide (împărţire în virgulă flotantă)
— *operaţie:* **A←A/B** şi **R←(R)+4**

Se împarte argumentul A la argumentul B, rezultatul depunîndu-se în locul argumentului A, deci:

[(R)+4, (R)+6]←[(R)+4, (R)+6]/[(R), (R)+2].

Operaţiile prezentate sînt adevărate numai în cazurile în care nu apar erori; tot în aceste cazuri, indicatorii de condiţii se poziţionează în modul următor:

N=1 dacă mantisa rezultatului este negativă; în rest, N=0;
Z=1 dacă rezultatul este zero; în rest, N=0;
V=0;
C=0.

2.7.3. CAZURILE DE EROARE

Sînt trei cazuri de eroare:

a) *depășire flotantă inferioară (underflow)*, adică rezultatul operației este mai mic decît 2^{-128}; în această situație în locul rezultatului se depune numărul zero exact, deci: [(R)+4, (R)+6]←0.

b) *depășire flotantă superioară (overflow)*, adică rezultatul operației este mai mare dcît 2^{-127}; în această situație cele două argumente rămîn nemodificate.

c) încercarea de împărțire prin zero; rezultatul coincide cu cel de la cazul b).

În caz de eroare se generează o întrerupere internă cu vectorul 244. Indicatorii de condiții vor fi poziționați în modul următor:

EROAREA	N	Z	V	C
UNDERFLOW	1	0	1	0
OVERFLOW	0	0	1	0
ÎMPĂRȚIRE PRIN ZERO	1	0	1	1

2.8. Tehnici de programare

2.8.1. TRATAREA ȘIRURILOR DE CARACTERE

Pentru tratarea șirurilor de caractere este comod să se utilizeze instrucțiuni pe byte cu autoincrementare și instrucțiunea SOB (vezi 2.4.2.1.) pentru decrementarea contorului de caractere și luarea unei decizii atunci cînd toate caracterele din șir au fost tratate. Exemplele următoare conțin programe simple scrise în mnemonice, putînd fi imediat transcrise în cod mașină.

a) *Transferul unui șir de N caractere (bytes) de la adresa ADR1 la adresa ADR2.*

```
A0 MOV #ADR1, R0
   MOV #ADR2, R1
   MOV #N, R2
A1 MOVB (R0)+, (R1)+
   SOB  R1, A1
   HALT
```

R0 (respectiv R1) conține adresa șirului emițător (respectiv receptor), adresă ce este incrementată cu 1 după fiecare byte transferat cu instrucțiunea MOVB de la adresa A1; R2 conține numărul de caractere de transferat, număr ce este decrementat după fiecare transfer; cînd (R1)=0 se intră în HALT, la panou afișîndu-se adresa de după HALT și conținutul registrului R0 (prima adresă după sfîrșitul șirului emițător).

b) *Compararea a două șiruri de caractere formate din N bytes, primul aflat la adresa ADR1, iar cel de-al doilea la adresa ADR2;* se afișează la panou adresa corespunzătoare primului șir la care apare diferență între caractere; dacă șirurile sînt identice se afișează la panou informația 177 777.

```
A0 MOV #ADR1, R0
   MOV #ADR2, R1
   MOV #N, R2
```

```
A2 CMPB (R0)+, (R1)+
   BNE  A1
   SOB  R2, A2
   MOV #177 777, R0
A1 HALT
```

c) *Se afişează la panou numărul de caractere cu codul xxx care se găsesc în şirul cu adresa de început ADR şi de lungime N.*

```
A0 CLR R0
   MOV #ADR, R1
   MOV #N, R2
A2 CMPB #xxx  (R1)
   BNE    A1
   INC    R0
A1 SØB  R2, A2
   HALT
```

2.8.2. MODURI DE TRANSMITERE A PARAMETRILOR UNEI SUBRUTINE

De cele mai multe ori, un program chemător trebuie să transfere unei subrutine apelate anumiţi *parametri* sau *argumente* (operanzi, date, adrese de operanzi sau date, etc.). În exemplele următoare se consideră trei argumente (A1, A2 şi A3), subrutina apelată (SUB) executînd următoarea operaţie: adună operanzii de la adresele A1 şi A2 şi depune rezultatul la adresa A3, fără a deteriora informaţia de la adresele A1 şi A2. Programul chemător începe la adresa A0, AR fiind adresa de revenire.

a) *Transmiterea parametrilor cu ajutorul registrelor generale:*

```
A0 MOV #A1, R0
   MOV #A2, R1
   MOV #A3, R2
   JSR PC, SUB1           cheamă subrutina SUB1
AR ______________________
SUB1 MOV(R0), R3
     ADD (R1), R3
     MOV R3, (R2)
     RTS PC               revenire în programul principal
```

Dezavantajul acestui mod de transmitere a parametrilor constă în necesitatea de a avea la dispoziţie suficiente registre generale libere; dacă registrele conţin informaţii utile ele vor trebui să fie salvate (pe stivă) înainte de încărcarea argumentelor şi vor fi reîncărcate după întoarcerea în programul chemător.

b) *Transmiterea imediată a argumentelor*

Se utilizează instrucţiunea JSR cu alt registru decît PC (vezi 2.4.3.1.); uzual este preferat R5; argumentele vor urma instrucţiunea JSR şi vor fi deci adresate de R5; în subrutină R5 va trebui să fie astfel utilizat încît la sfîrşit să conţină adresa de revenire (AR):

```
A0 JSR R5, SUB2
   A1
   A2
   A3
AR . . .   . . . . . .
```

```
SUB2 MOV  @(R5)+, R0
     ADD  @(R5)+, R0
     MOV R0, @(R5)+
     RTS R5
```

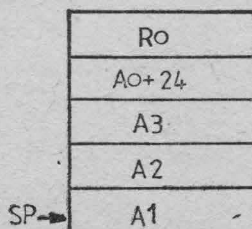
Acest mod este comod și nu utilizează decît registrul general R5 (indiferent de numărul parametrilor); argumentele trebuiesc exploatate în ordine, pentru ca la sfîrșitul subrutinei R5 să conțină adresa de revenire în programul chemător.

c) *Transmiterea argumentelor cu ajutorul stivei*

Înainte de apelarea subrutinei se depun argumentele pe stivă; dacă argumentele sînt utilizate o singură dată și se cunoaște ordinea folosirii lor, programul poate fi optimizat:

```
A0 MOV R0, —(SP)              salvare RO pe stivă.
   MOV 16(PC), —(SP)          depunere adresă de revenire
   MOV #A3, —(SP)             depunere argumente
   MOV #A2, —(SP)
   MOV #A1, —(SP)
   JMP SUB3
A0+24  MOV (SP)+, R0          reîncărcare R0 și eliberare stivă
```

Stiva va avea configurația:




```
SUB3 MOV       @(SP)+, R0
     ADD       @(SP)+, R0
     MOV R0,   @(SP)+         revenire la adresa A0+24
     RTS PC
```

Dacă se dorește o apelare oarecare a argumentelor se va utiliza modul de adresare indexat cu SP:

```
A0  MOV R0, —(SP)
    MOV #A1, —(SP)
    MOV #A2, —(SP)
    MOV #A3, —(SP)
    JSR PC, SUB4              eliberare stivă
AR  ADD  #6, SP
    MOV  (SP)+, R0
```

```
A2 CMPB (R0)+, (R1)+
   BNE A1
   SOB R2, A2
   MOV #177 777, R0
A1 HALT
```

c) *Se afişează la panou numărul de caractere cu codul xxx care se găsesc în şirul cu adresa de început ADR şi de lungime N.*

```
A0 CLR R0
   MOV #ADR, R1
   MOV #N, R2
A2 CMPB #xxx (R1)
   BNE  A1
   INC  R0
A1 SØB R2, A2
   HALT
```

2.8.2. MODURI DE TRANSMITERE A PARAMETRILOR UNEI SUBRUTINE

De cele mai multe ori, un program chemător trebuie să transfere unei subrutine apelate anumiţi *parametri* sau *argumente* (operanzi, date, adrese de operanzi sau date, etc.). În exemplele următoare se consideră trei argumente (A1, A2 şi A3), subrutina apelată (SUB) executînd următoarea operaţie: adună operanzii de la adresele A1 şi A2 şi depune rezultatul la adresa A3, fără a deteriora informaţia de la adresele A1 şi A2. Programul chemător începe la adresa A0, AR fiind adresa de revenire.

a) *Transmiterea parametrilor cu ajutorul registrelor generale:*

```
A0 MOV #A1, R0
   MOV #A2, R1
   MOV #A3, R2
   JSR PC, SUB1          cheamă subrutina SUB1
AR ____________________
SUB1 MOV (R0), R3
     ADD (R1), R3
     MOV R3, (R2)
     RTS PC              revenire în programul principal
```

Dezavantajul acestui mod de transmitere a parametrilor constă în necesitatea de a avea la dispoziţie suficiente registre generale libere; dacă registrele conţin informaţii utile ele vor trebui să fie salvate (pe stivă) înainte de încărcarea argumentelor şi vor fi reîncărcate după întoarcerea în programul chemător.

b) *Transmiterea imediată a argumentelor*

Se utilizează instrucţiunea JSR cu alt registru decît PC (vezi 2.4.3.1.); uzual este preferat R5; argumentele vor urma instrucţiunea JSR şi vor fi deci adresate de R5; în subrutină R5 va trebui să fie astfel utilizat încît la sfîrşit să conţină adresa de revenire (AR):

```
A0 JSR R5, SUB2
   A1
   A2
   A3
AR . . . . . . . . .
```

```
SUB2 MOV  @(R5)+, R0
     ADD  @(R5)+, R0
     MOV R0, @(R5)+
     RTS R5
```

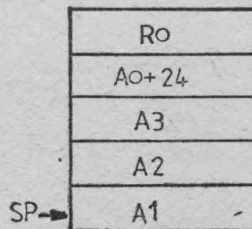
Acest mod este comod și nu utilizează decît registrul general R5 (indiferent de numărul parametrilor); argumentele trebuiesc exploatate în ordine, pentru ca la sfîrșitul subrutinei R5 să conțină adresa de revenire în programul chemător.

c) *Transmiterea argumentelor cu ajutorul stivei*

Înainte de apelarea subrutinei se depun argumentele pe stivă; dacă argumentele sînt utilizate o singură dată și se cunoaște ordinea folosirii lor, programul poate fi optimizat:

```
A0 MOV R0, —(SP)              salvare RO pe stivă.
   MOV 16(PC), —(SP)          depunere adresă de revenire
   MOV #A3, —(SP)             depunere argumente
   MOV #A2, —(SP)
   MOV #A1, —(SP)
   JMP SUB3
A0+24 MOV (SP)+, R0           reîncărcare R0 și eliberare stivă
```

Stiva va avea configurația:

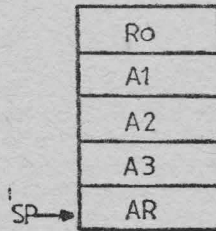



```
SUB3 MOV      @(SP)+, R0
     ADD      @(SP)+, R0
     MOV R0,  @(SP)+          revenire la adresa A0+24
     RTS PC
```

Dacă se dorește o apelare oarecare a argumentelor se va utiliza modul de adresare indexat cu SP:

```
A0  MOV R0, —(SP)
    MOV #A1, —(SP)
    MOV #A2, —(SP)
    MOV #A3, —(SP)
    JSR PC, SUB4              eliberare stivă
AR  ADD  #6, SP
    MOV  (SP)+, R0
```

În momentul apelării subrutinei, stiva conține:

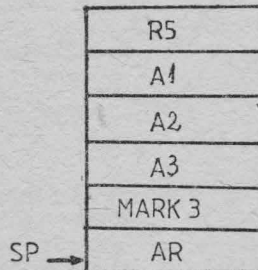



```
SUB4 MOV      @6(SP), R0
     ADD      @4(SP), R0
     MOV R0,  @2(SP)
     RTS PC
```

d) *Transmiterea argumentelor cu ajutorul instrucțiunii MARK (vezi 2.4.3.3.)*

```
A0  MOV R5, —(SP)              salvare R5, utilizat de MARK
    MOV # A1, —(SP)            depunere argumente pe stivă
    MOV # A2, —(SP)
    MOV # A3, —(SP)
    MOV # MARK 3, —(SP)        depunere MARK
    MOV SP, R5                 adresă MARK → R5
    JSR PC, SUB5
AR   ⋮                         (adresa de revenire)
```

Stiva conține:



Subrutina va folosi modul de adresare indexat cu SP pentru apelarea argumentelor, conținutul lui SP trebuind să rămînă neschimbat:

```
SUB5 MOV      @10(SP), R0
     ADD      @6(SP), R0
     MOV R0,  @4(SP)
     RTS R5                 revenire și eliberare stivă.
```

Observație: în cazul lucrului cu subrutinele trebuie inițializat în prealabil SP (definirea vîrfului stivei), deoarece instrucțiunile JSR, RTS și MARK utilizează stiva.

2.8.3. IMBRICAREA ÎNTRERUPERILOR ȘI A SUBRUTINELOR

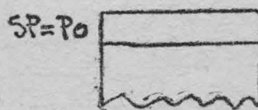
O întrerupere este similară cu apelarea unei subrutine, cu excepţia că ea este iniţiată în special de hardware (spre deosebire de subrutină care este apelată prin program, deci, software). O întrerupere externă poate apărea oricînd, dar microprogramul de emulare nu o ia în consideraţie decît la sfîrșitul execuţiei unei instrucţiuni.

Întreruperile externe (de la periferice) sînt utilizate pentru a reduce timpul de așteptare al procesorului. Acesta nu va mai trebui să testeze starea perifericului, ci va putea lucra alt program pînă la primirea unei întreruperi ce va marca încheierea execuţiei funcţiei perifericului. După cum s-a arătat în capitolul 2.6., întreruperea constă în salvarea stării program întrerupte și saltul la un subprogram de tratare a acesteia (adresa de început a subprogramului aflîndu-se la adresa-vector corespunzătoare întreruperii respective).

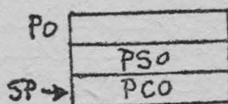
Dacă înainte de încheierea tratării unei întreruperi apare o altă întrerupere prioritară, programul de tratare a primei întreruperi va fi de asemenea întrerupt, sărindu-se la rutina corespunzătoare întreruperii prioritare. Procesul se va repeta ori de cîte ori va apare o nouă întrerupere prioritară, pe stivă fiind salvate în ordine stările program succesiv întrerupte (la care se va reveni). Rezultă că întreruperile pot fi imbricate, tot așa cum pot fi imbricate și subrutinele; de asemenea, se pot intercala cereri de întrerupere cu apeluri de subrutine, adresele de revenire fiind salvate pe stivă în ordine. Utilizînd instrucţiunile RTI și RTS se va reveni automat în programele întrerupte.

Exemplul următor arată evoluţia stivei în cazul imbricării întreruperilor și a apelurilor de subrutine.

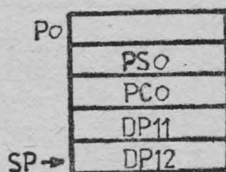
1. Este activ programul 0; SP este iniţializat la valoarea P0 (vîrful stivei).



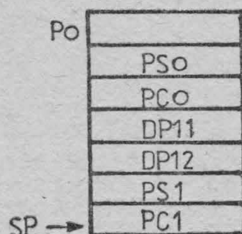
2. Este întrerupt programul 0, salvîndu-se pe stivă starea sa (PS0) și contorul de program (PC0), necesare la revenire; este lansat programul 1 de tratare a întreruperii.



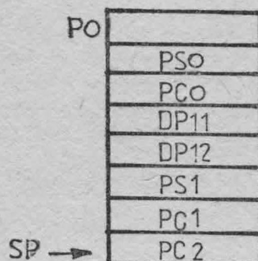
3. Programul 1 utilizează stiva pentru a memora temporar două date: DP11 și DP12.



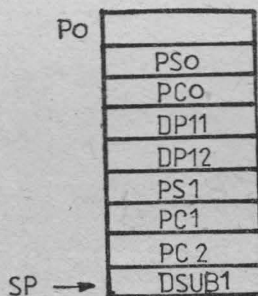
4. Programul 1 este întrerupt de o nouă întrerupere (prioritară) la starea (salvată) PS1 şi PC1; este lansat programul 2 de tratare a noii întreruperi.



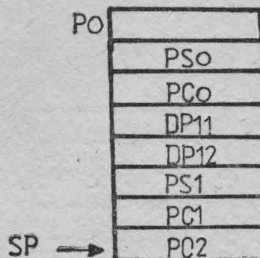
5. Programul 2 apelează subrutina SUB1, salvîndu-se pe stivă adresa de revenire PC2.



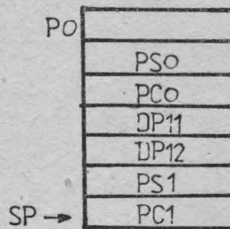
6. Subrutina SUB1 memorează temporar pe stivă data DSUB1.



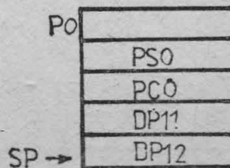
7. Subrutina SUB1 preia data DSUB1, eliberînd stiva.



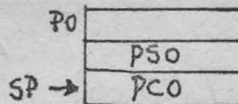
8. Se revine în programul 2, cu o instrucțiune RTS PC, la adresa de revenire PC2.



9. Programul 2 se încheie cu o instrucțiune RTI care realizează revenirea în programul 1 (prin forțarea stării program PS1 și a controlului de program PC1); programul 1 continuă din punctul în care a fost întrerupt.



10. Programul 1 preia datele DP11 și DP12 eliberînd stiva.



11. Programul 1 se încheie cu o instrucțiune RTI, revenindu-se în programul 0 (principal) la adresa PC0, cu starea PS0; programul 0 continuă din punctul în care a fost întrerupt. Stiva este eliberată complet.



Întreruperile care pot apărea în timpul execuției unui program pot fi de două tipuri:

— *întreruperi externe*, de la dispozitivele periferice, care indică terminări normale sau anormale ale operațiilor de intrare-ieșire;

— *întreruperi interne* care marchează detecția unor erori majore sau care sînt generate în urma execuției unor instrucțiuni speciale.

Diferența dintre cele două tipuri de întreruperi este următoarea: întreruperile interne nu pot fi mascate (sau invalidate), ele declanșîndu-se imediat în primul *punct întreruptibil* întîlnit în microprogramul de emulare de la apariția condiției de întrerupere; acest punct întreruptibil este plasat, în general, între execuțiile a două instrucțiuni consecutive (v. 1.1.3.). Întreruperile externe sînt organizate pe nivele de prioritate și pot fi mascate; o întrerupere externă va fi acceptată în punctul întreruptibil numai dacă nivelul său de prioritate este strict mai mare decît nivelul de prioritate al programului ce trebuie întrerupt (nivel ce este codificat în zona NIT din registrul de stare program PSW, vezi 2.1.1.2.).

Vectorii de întrerupere sînt prelucraţi prin intermediul microprogramului de emulare, fiind relocaţi dacă dispozitivul management este activat. În continuare sînt prezentate cauzele întreruperilor interne şi vectorii corespunzători:

NR.	CAUZĂ ÎNTRERUPERE INTERNĂ	VECTOR (OCTAL)
1.	adresare impară nepermisă	4
2.	eroare management	250
3.	time-out pe bus (adresă nerecunoscută)	4
4.	eroare paritate la memoria centrală	114
5.	cădere tensiune alimentare	24
6.	revenire tensiune alimentare	24
7.	instrucţiune ilegală	4
8.	instrucţiune rezervată	10
9.	depăşire stivă (STACK OVERFLOW)	4
10.	poziţionare bit T din PSW	14
11.	instrucţiunea BPT	14
12.	instrucţiunea IOT	20
13.	instrucţiunea EMT	30
14.	instrucţiunea TRAP	34
15.	eroare sau condiţie specială în virgulă mobilă	244

2.8.4. CORUTINE

Corutinele sînt două secţiuni de program care se pot chema una pe cealaltă (se mai numesc şi rutine interactive). Trecerea de la o rutină la cealaltă se poate face uşor utilizînd un caz particular al instrucţiunii JSR:

JSR PC, @(R6)+

Instrucţiunea este utilizată atît pentru chemarea unei rutine cît şi pentru revenirea în altă rutină.

În prima rutină se va depune pe stivă adresa de început a rutinei a doua (SUB2):

MOV # SUB2, —(SP)

În rutina a doua se va intra cu instrucţiunea:

JSR PC, @(R6)+

care execută următoarea secvenţă de operaţii:

a) se preia de pe stivă SUB2 (adresat de R6) şi apoi R6 este autoincrementat;

b) R6 este autodecrementat şi apoi se depune pe stivă adresa de revenire în prima corutină (PC actualizat);

c) SUB2 este introdus în PC, dîndu-se astfel controlul corutinei a doua.

După ce se execută o serie de operaţii, se va reveni în corutina întîi cu aceeaşi instrucţiune:

JSR PC, @(R6)+;

de data aceasta se va salva pe stivă adresa de revenire în corutina a doua. Procesul se poate repeta, cele două rutine chemîndu-se una pe cealaltă în anumite momente de timp, execuţia lor reluîndu-se de fiecare dată din punctul în care au fost întrerupte. Se observă că SP-ul „bate pasul pe loc", adresele de revenire în corutine salvîndu-se în acelaşi cuvînt de pe stivă.

Trebuie avut grijă ca, în momentul saltului în cealaltă corutină, SP-ul să fie poziţionat corespunzător (să corespundă adresei de revenire în corutina apelată, adresă ce a fost salvată anterior).

Diferențele dintre corutine și subrutine sînt următoarele (fig. 2—8):

— o subrutină poate fi considerată subordonată unui program principal sau altei subrutine chemătoare; corutinele sînt considerate de același nivel, fiecare putînd să cheme altă corutină cînd a completat procesul curent;

— o subrutină apelată este executată în întregime, de la început la sfîrșit, o corutină este executată din punctul aflat după ultima chemare a altei corutine; deci, în timp ce o subrutină apelată execută aceleași operații, o corutină nu va executa aceleași operații la fiecare apelare a ei.

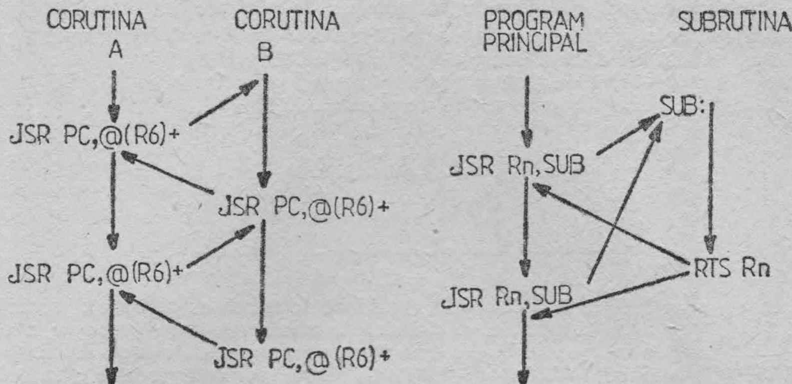


FIG. 2–8. CORUTINE ȘI SUBRUTINE.

Corutinele sînt utilizate atunci cînd două programe trebuie să fie coordonate în execuție, existînd o relație de simetrie între ele.

3. EXTENSII

3.1. Controlorul de paritate al memoriei centrale

3.1.1. PARITATE

Memoria minicalculatoarelor **INDEPENDENT** și **CORAL** este de tip MOS-DINAMIC (cu regenerarea informației), realizată tehnologic pe *planuri* (plăci) de 32 KO, 64 KO, 128 KO, 256 KO, și chiar 1 MO. Unitatea adresabilă este *byte-ul (octetul)*, adresele crescînd cu 1 din octet în octet (v. 1.3.1.). Operația de scriere în memorie se poate executa atît pe byte, cît și pe cuvînt; în cazul lecturii se citește oricum tot cuvîntul de la adresa pară respectivă (forțîndu-se la 0 bitul cel mai puțin semnificativ), urmînd ca procesorul să utilizeze, eventual, numai un byte.

Fiecărui byte i se asociază cîte un bit de paritate (de control); deci, din punctul de vedere al memoriei centrale, cuvintele vor avea cîte 18 biți (16 de date și 2 de paritate). Se utilizează *paritatea impară* (ODD PARITY), adică unui octet îi corespunde un bit a cărui valoare este astfel încît numărul de 1 din ansamblul octet și bit de paritate să fie impar. Deci:

— dacă un octet are un număr par de 1 (0 este considerat număr par) atunci bitul de paritate va fi 1;

— dacă un octet are un număr impar de 1, atunci bitul său de paritate va fi 0.

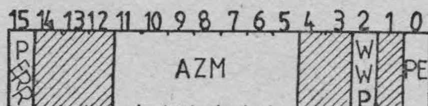
Bitul de paritate e calculat și memorat într-o locație specială la operația de scriere în memorie. În cazul operației de citire din memorie, bitul de paritate este recalculat și comparat cu cel memorat în locația corespunzătoare octetului citit; în

caz de coincidență a bitului calculat cu cel memorat se consideră că octetul s-a păstrat corect în memorie; în caz de necoincidență este raportată eroarea corespunzătoare de paritate și se va executa o întrerupere internă cu vectorul 114.

3.1.2. REGISTRUL DE STARE AL CONTROLORULUI DE PARITATE

Eventualele erori de paritate nu sînt raportate direct procesorului, ci prin intermediul unui dispozitiv special numit *controlor de paritate;* acesta poate fi validat sau invalidat (raportînd sau neraportînd eroarea de paritate detectată).

Starea controlorului de paritate este descrisă de registrul corespunzător de control și stare (CSR) cu adresa selectabilă între 172 100 și 172 136 (în cazul adreselor pe 18 biți: 772 100—772 136). Toate pozițiile acestui registru sînt READ/WRITE (pot fi scrise și citite), permițîndu-se astfel verificarea lui.



p.b.0: PE = PARITY ENABLE (validare controlor paritate); scriind 1 în această poziție se validează trimiterea spre procesor a erorii de paritate detectate; este șters prin scriere 0 sau cu INIT.

p.b.2: WWP = WRITE WRONG PARITY (scrie paritate proastă); indică modul de calcul al bitului de paritate:
0=paritate impară (corectă);
1=paritate pară (incorectă).
Acest indicator influențează modul de calcul al bitului de paritate numai prin forțarea unei parități pare, incorecte (inversă celei impare, corecte). Bitul poate fi setat numai prin scriere 1 în poziția corespunzătoare din registru; este șters prin scriere cu 0 sau cu INIT.

p.b.5—11: AZM = ADRESĂ ZONĂ MEMORIE în care a apărut o eroare de paritate; în aceste poziții se memorează biții cei mai semnificativi ai adresei de memorie la care a fost detectată o eroare de paritate (este vorba de biții 11—17 ai adresei, deci se poate localiza zona de 1K cuvînt sau de 2KO în care a apărut eroarea). Pozițiile acestea pot fi încărcate atît hardware (în cazul detecției unei erori de paritate) cît și prin program (pentru verificarea registrului).

p.b.15: PERR = PARITY ERROR (eroare paritate); indicatorul este poziționat în cazul detectării unei erori de paritate; dacă și p.b.0 (PE) este 1, eroarea este raportată la procesor pe o linie de bus specială. Indicatorul poate fi șters prin scriere cu 0 sau cu INIT.

Observații: minicalculatoarele INDEPENDENT au un singur controlor de paritate pentru toată memoria, cu un singur registru de stare și control, cu adresa 772 100 (localizat pe placa TLP-100). Minicalculatoarele CORAL au cîte un controlor de paritate pentru fiecare plan (placă) de memorie (adresele registrelor corespunzătoare fiind succesive: 772 100, 772 102 etc.).

3.1.3. VERIFICAREA CONTROLULUI DE PARITATE

Verificarea controlorului de paritate (raportarea unei erori și memorarea corectă a adresei zonei de memorie respectivă) se face prin forțarea unei parități pare (incorecte). Se procedează în modul următor:

a) *ETAPA I:*

— se șterge registrul de stare și control al controlorului (CSR);
— se activează controlorul scriind #1 în CSR (paritate impară);

— se depune la o adresă de memorie (de exemplu 077776) o cantitate oarecare (de exemplu 012345);

— se examinează adresa respectivă (077776) pentru a provoca o citire din memorie și deci detecția unei eventuale erori de paritate; datorită faptului că WWP=0 (paritate corectă) nu trebuie să apară eroare;

— se examinează registrul CSR; va trebui să conțină informația 000 001.

b) *ETAPA a II-a:*

— se șterge registrul CSR;

— se activează controlorul scriind # 5 în CSR (paritate pară, incorectă);

— se depune la adresa 077 776 o cantitate oarecare;

— se examinează adresa 077 776; datorită faptului că biții de paritate au fost scriși negați față de valorile corecte, trebuie să apară eroare (să se aprindă indicatorul luminos de eroare generală de pe panou);

— se examinează registrul CSR; el trebuie să conțină 100 745 înainte de inițializare, respectiv 000 740 după inițializare (AZM=000 111 în cazul adresei 077 776).

Observații: procedura se poate executa de la panoul de comandă, cu ajutorul emulatorului de consolă sau scriind un program corespunzător (atenție la cazul în care apare eroare; după cum s-a arătat în capitolul 2.6. va avea loc o întrerupere internă cu vectorul 4).

3.2. Managementul de memorie

3.2.1. FUNCȚIILE MANAGEMENTULUI

După cum s-a arătat și în capitolul 1.2.2., procesoarele minicalculatoarelor **CORAL** și **INDEPENDENT** pot genera adrese de instrucțiuni și operanzi pe 16 biți (un cuvînt); în această situație se poate adresa direct o memorie cu capacitatea maximă de 64 KO. Pentru adresarea unei memorii extinse la 256 KO sînt necesare adrese pe 18 biți. Trecerea de la adresa logică pe 16 biți pe care o prezintă procesorul la adresa fizică pe 18 sau chiar mai mulți biți se numește *relocare;* operația este executată printr-o procedură de bazare, realizată cu ajutorul unui dispozitiv special numit *management de memorie.*

De fapt, busurile calculatoarelor CORAL și INDEPENDENT au 22 linii de adresă (A00÷A21) ceea ce va permite o extensie ulterioară a memoriei pînă la maxim 4MO (bineînțeles și cu modificarea corespunzătoare a managementului).

Totodată, filozofia unibus impune rezervarea adreselor din ultimii 8KO (4 K cuvinte) de memorie pentru necesitățile sistemului de intrare/ieșire (adrese de serviciu). Pentru ca adresele de serviciu să fie plasate în ultimii 8KO de memorie și dacă managementul nu există sau este inactivat (OFF LINE), toate adresele logice cuprinse între 160 000 și 177 777 sînt convertite în adrese pe 18 biți cu biții de pondere superioară (16 și 17) forțați la 1 logic (obținîndu-se astfel adresele 760000—777777). De exemplu, adresa pe 16 biți 177 560 (care, cu managementul inactiv, va apela un registru dintr-un cuplor) va fi convertită intern în adresa 777 560.

Blocul *MANAGEMENT* rezolvă 2 probleme principale:

1) *relocarea adreselor:* trecerea de la adresa logică pe 16 biți (prezentată de procesor) la adresa fizică pe 18 biți capabilă să adreseze o memorie de maxim 256 KO (128 K cuvinte). Operația este executată prin însumarea adresei logice cu o așa numită *constantă de relocare,* memorată într-un registru special și care constituie, practic, adresa de început a unei anumite pagini de memorie.

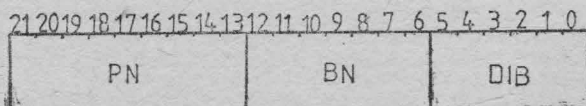
2) *împărțirea memoriei în pagini, controlul și protecția lor.*

Se introduc următoarele noțiuni:

a) **BLOC:** zonă de memorie de lungime fixă (32(10) cuvinte sau 64(10) bytes); adresele din interiorul blocului sînt determinate de cei 6 biți mai puțin semnificativi ai adresei fizice, putînd lua valorile octale: 00 — 77.

b) **PAGINĂ:** zonă de memorie de lungime variabilă, putînd avea între 1 și 128(10) blocuri; o adresă din interiorul unei pagini este dată de ultimii 13 biți mai puțin semnificativi ai adresei fizice.

Astfel, o adresă de bus (de exemplu pe 22 de biţi) poate fi împărţită în următoarele zone:



PN = PAGE NUMBER (numărul paginii din memorie);
BN — BLOCK NUMBER (numărul blocului din pagină);
DIB = DISPLACEMENT IN BLOCK (deplasamentul în interiorul blocului).

O pagină poate avea între 32 şi 4 096 de cuvinte, adresele dintr-o pagină de lungime maximă putînd lua valori între 00000 şi 17777.

Definind paginile, controlînd modul de acces la aceste pagini (protecţie) precum şi repartizarea lor dinamică în memorie (relocare), menagementul facilitează lucrul în multiprogramare (permiţînd existenţa mai multor programe active simultan în memoria calculatorului).

Procesorul poate opera în 2 moduri: **Kernel şi User.** În Kernel, programul are controlul complet asupra sistemului şi poate executa orice instrucţiune; în acest mod este executat monitorul şi programele de supervizare. În User, programului i se interzice execuţia anumitor funcţii ce ar putea modifica starea procesorului, a programului Kernel, sau ar putea determina oprirea procesorului sau utilizarea spaţiilor de memorie atribuite altor utilizatori.

În procedura de multiprogramare sînt rezidente în memorie monitorul, programele de supervizare, precum şi mai multe programe utilizator (executate în modul USER).

Programele de supervizare au următoarele roluri:
— controlează şi execută programele utilizatorilor;
— alocă pagini de memorie utilizatorilor în funcţie de posibilităţi şi necesitatea acestora;
— controlează operaţiile cu echipamentele periferice (gestiunea fişierilor);
— protejează utilizatorii între ei precum şi integritatea sistemului.

Din punctul de vedere al managementului de memorie, există următoarele situaţii:

a) *minicalculatoare fără management;* acestea pot avea o memorie internă de maxim 64 KO; este cazul minicalculatorului CORAL 4001.

b) *minicalculatoare cu management simplificat;* acestea pot avea o memorie internă extinsă la 256 KO sau chiar mai mult (4MO); este cazul minicalculatoarelor CORAL 4011/4030 şi INDEPENDENT-100/102F.

c) *minicalculatoare cu management complet;* capacitatea memoriei centrale nu este influenţată, dar sînt completate funcţiile managementului (de exemplu: se pot defini pagini (zone) speciale pentru instrucţiuni şi date, se permite lucrul şi într-un al treilea mod *(SUPERVIZOR)* etc.). Cu un astfel de management sînt echipate variantele dezvoltate din familia minicalculatoarelor româneşti (de exemplu INDEPENDENT 102 F).

3.2.2. MANAGEMENTUL SIMPLIFICAT

3.2.2.1. **Registrele de pagină activă.** Fiecărei pagini active la un moment dat îi corespunde cîte un *dublu registru de pagină activă* (APR=ACTIVE PAGE REGISTER). În total există 16 astfel de registre, 8 pentru modul de lucru Kernel şi 8 pentru modul de lucru User (fig. 3-1).

Rezultă că, la un moment dat, numai cîte 8 pagini de memorie (maxim 64 KO) sînt active pentru fiecare mod de lucru. Pentru utilizarea şi a altor pagini disponibile este necesară reîncărcarea corespunzătoare ale registrelor APR. După cum se va vedea mai departe, toate registrele managementului pot fi apelate prin program,

avînd alocate adrese de serviciu (din ultimii 8 KO rezervaţi sistemului de intrare/ieşire).

Managementul simplificat acceptă numai două moduri de lucru: Kernel (codificat 00 în PSW 14-15) şi User (11); modurile 01 şi 10 sînt ilegale, orice acces la memorie sub un astfel de mod fiind abandonat.

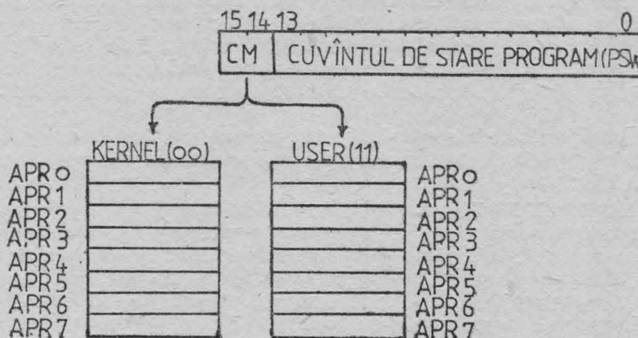
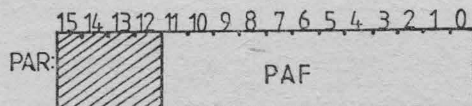


Fig. 3-1. REGISTRELE DE PAGINĂ ACTIVĂ.

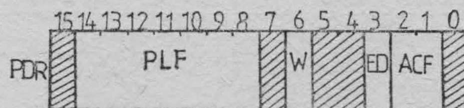
Fiecare APR conţine informaţii ce definesc şi descriu pagina respectivă. Un APR este alcătuit din două registre distincte:

a) *registrul de adresă pagină* (PAR=PAGE ADRESS REGISTER)



p.b.0—11: PAF=PAGE ADRESS FIELD (cîmpul de adresă pagină); registrul conţine adresa de bază a paginii (constanta de relocare). Conţinutul registrului nu poate fi modificat decît prin scriere prin program, toate poziţiile fiind READ/WRITE. Poziţiile 12—15 sînt neutilizate în cazul folosirii a numai 18 linii de adresă (ele sînt rezervate pentru extinderea ulterioară a lungimii adresei în vederea măririi capacităţii memoriei centrale). Pentru procedura de relocare a adresei vezi paragraful 3.2.5.1.

b) *registrul descriptor de pagină* (PDR=PAGE DESCRIPTOR REGISTER).



Toate poziţiile acetui registru pot fi scrise şi citite prin program cu excepţia bitului 6 (W) care este READ ONLY (poate fi numai citit). Poziţiile registrului nu sînt afectate de INIT.

p.b.1—2: ACF = ACCES CONTROL FIELD (modul de acces la memorie); poziţiile codifică cheile de acces la memorie (modul în care este permis un acces în pagina respectivă). Violarea modului de acces duce la abortarea operaţiei şi semnalizarea erorii respective. În caz de operaţie permisă nu apare nici o întrerupere.

ACF 2 1

0 0 — nerezident (abort la orice acces în pagină);

0 1 — citire numai (abort la scriere în pagină);

1 0 — neutilizat (abort la orice acces);
1 1 — scriere/citire (este permis orice acces în pagină). ACF este scris prin program, definindu-se astfel cheia de acces la pagină.

p.b.3: ED = EXPANSION DIRECTION (direcţia de dezvoltare a paginii):
ED=0 indică faptul că pagina se dezvoltă pornind de la adresa sa minimă (zero relativ) spre adrese crescătoare (lungimea paginii creşte prin adăugarea de blocuri cu adrese relative din ce în ce mai mari);
ED=1 indică faptul că pagina se dezvoltă pornind de la adresa sa maximă (considerînd pagina de lungime maximă) spre adrese descrescătoare (lungimea paginii creşte prin adăugarea de blocuri cu adrese relative din ce în ce mai mici).
Bitul ED este scris sub controlul programului, definindu-se astfel direcţia de dezvoltare a paginii. Reprezentarea unei pagini în memorie este arătată în paragraful 3.2.5.2.

p.b.6: W = WRITEN INTO (pagină scrisă); indicatorul W=1 arată faptul că pagina respectivă a fost scrisă (s-a scris cel puţin un cuvînt în ea) din momentul în care a fost definită. Indicatorul este şters automat la orice scriere în registrul PAR sau PDR ataşat paginii respective (redefinirea paginii); poate fi setat numai de logica de control a managementului. Acest bit este folosit pentru a determina ce pagini de memorie au fost modificate de un utilizator; dacă W=1 rezultă necesitatea de a salva pagina (pe disc) în forma curentă (în cazul în care intervine şi un alt utilizator prioritar); dacă W=0 pagina nu mai trebuie salvată sau, chiar mai mult, poate fi definită ca altă pagină (dacă este necesar).

p.b.8—14: PLF = PAGE LENGHT FIELD (lungime pagină); această cantitate pe 7 biţi indică numărul de blocuri ale paginii, deci specifică lungimea paginii în blocuri de cîte 32 de cuvinte. PLF poate lua valori între 0 şi 177(8), ceea ce determină o lungime de pagină între 1 şi 128(10) blocuri. Rezultă că:

$$PLF=NB-1 \text{ sau } NB=PLF+1,$$

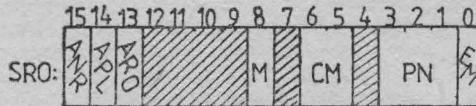
unde NB=numărul de blocuri ale paginii. PLF este scris prin program, definindu-se astfel lungimea paginii. Adresa cu care se încearcă un acces la memorie trebuie să nu depăşească lungimea definită (zona permisă) a paginii; în caz contrar operaţia este abandonată şi se poziţionează un indicator de eroare. Pentru reprezentarea paginii în funcţie de lungimea ei şi direcţia de dezvoltare vezi paragraful 3.2.5.2.

În total există 16 registre PAR şi 16 registre PDR, cîte 8 pentru fiecare mod de lucru; adresele acestor registre sînt următoarele:

KERNEL			USER		
i	PAR i	PDR i	i	PAR i	PDR i
0	772340	772300	0	777640	777600
1	772342	772302	1	777642	777602
2	772344	772304	2	777644	777604
3	772346	772306	3	777646	777606
4	772350	772310	4	777650	777610
5	772352	772312	5	777652	777612
6	772354	772314	6	777654	777614
7	772356	772316	7	777656	777616

3.2.2.2. **Registrele de stare.** O eroare detectată de management provoacă o întrupere internă cu vectorul 250. Managementul simplificat are două registre de stare (SR0 şi SR2) utilizate pentru a determina tipul erorii şi adresa la care a apărut aceasta.

a) *Registrul de stare 0* (SR0=STATUS REGISTER 0)
Adresa: 777 572



p.b.0: EM = ENABLE MANAGEMENT (validare funcții management); cînd această poziție este 1, funcțiile managementului sînt validate (MMG ON), adică:
— toate adresele prezentate de procesor pentru operațiile de intrare-ieșire sînt relocate conform procedurii prezentate în paragraful 3.2.5.1.;
— încercarea de a viola cheia de acces în pagina respectivă, precum și depășirea lungimii paginii, va provoca abandonarea operației și setarea unor indicatori de eroare corespunzători.
Dacă bitul este 0 managementul este dezactivat (OFF), neavînd loc nici relocarea și nici protecția adreselor (adresa logică, pe 16 biți, prezentată de procesor coincizînd cu adresa fizică de bus). Bitul EM poate fi scris sub controlul programului, definind astfel starea managementului. Poate fi de asemenea citit. Este șters cu INIT.

p.b.1—3: PN = PAGE NUMBER (număr pagină); cîmpul conține numărul paginii de memorie adresată la ultima operație de intrare-ieșire. Pozițiile sînt READ ONLY, ele neputînd fi scrise decît hardware cu biții cei mai semnificativi ai adresei logice (nerelocate); scrierea are loc numai dacă s-a accesat o pagină de memorie și dacă nu există vreo eroare de management memorată. În momentul în care apare eroare (abort) pozițiile 1—3 „îngheață" memorînd numărul paginii în care a apărut eroarea; acest număr va fi folosit ulterior de rutina de identificare a paginii în care s-a produs abortul.

p.b.5—6: CM = CURRENT MOD (mod curent de lucru); acești biți conțin modul de lucru curent (PSW 14—15) sub care s-a făcut ultimul acces la memorie; biții sînt READ ONLY, ei putînd fi scriși numai hardware în aceleași condiții ca și biții 1—3 (PN); rezultă că în cazul unui abort ei vor „îngheța" indicînd modul curent sub care a apărut eroarea (Kernel=00, User=11).

p.b.8: M = MAINTENANCE (întreținere); dacă acest bit este *1*, sînt forțate funcțiile de relocare și protecție ale managementului chiar dacă acesta este inactivat (p.b.0=0); funcțiile sînt forțate numai în cazul adresării unui operand destinație de către o instrucțiune (sub acțiunea emulatorului). Rezultă că în cazul execuției unei instrucțiuni se va reloca numai adresa operandului destinație. De exemplu, pentru execuția instrucțiunii MOV SS, DD (cei doi operanzi fiind în memorie, neindexați și neindirectați) trebuiesc făcute 3 accese la memorie:
1) citirea instrucțiunii;
2) citirea operandului sursă;
3) scrierea sursei la destinație;
dacă p.b.0=1 (EM=1) toate cele 3 operații se vor executa cu adrese relocate; dacă p.b.0=0 și p.b.8=1 (M=1) atunci numai cea de a 3-a operație se va executa cu adresa relocată; în felul acesta se poate verifica ușor funcția de relocare a managementului. Bitul este READ-WRITE, deci poate fi poziționat prin program.

Biții 13, 14 și 15 sînt indicatorii de eroare (abort management):

p.b.13: ARO = ABORT READ ONLY (violare pagină citire numai); eroarea apare dacă se încearcă o scriere într-o pagină cu cheia de acces 01.

p.b.14: APL = ABORT PAGE LENGHT (depășire lungime pagină); eroarea apare dacă numărul blocului din adresa logică prezentată de procesor depășește lungimea paginii indicată de cîmpul PLF din registrul descriptor PDR (ținînd cont, bineînțeles, și de direcția de dezvoltare a paginii).

p.b.15: ANR = ABORT NONRESIDENT (violare pagină nerezidentă (nedefinită)); eroarea apare la orice încercare de a adresa o pagină nerezidentă, deci cu cheia de acces 00 sau 10. Tot acest indicator de eroare este poziționat și dacă se în-

cearcă un acces la memorie cu mod ilegal în PSW (PSW 14—15=01 sau PSW 14—15=10).

Apariția unei erori de management provoacă o întrerupere cu vectorul 250 (este vorba de întrerupere internă, deci nu poate fi mascată). Pozițiile 13, 14 și 15 din SR0 pot fi poziționate hardware (la detectarea erorii respective), dar ele pot fi scrise și prin program (deci biții sînt READ-WRITE); setînd biții 13—15 sub controlul programului nu se declanșează întreruperea cu vectorul 250, scrierea fiind utilizată numai pentru a verifica aceste poziții ale registrului.

Observație: tot registrul SR0 este șters cu INIT. Biții de abort (SR0—13—15) pot fi șterși prin scriere cu 0.

b) *Registrul de stare 2* (SR2=STATUS REGISTER 2)
Adresa:777 576

Registrul SR2 are lungimea de un cuvînt și este încărcat hardware cu adresa logică prezentată de procesor la începutul fiecărei citiri de instrucțiune din memorie (FETCH); dacă apare o eroare de management conținutul SR2 „îngheață" pînă ce indicatorii de eroare din SR0 sînt șterși; în acest fel se conservă adresa instrucțiunii care a provocat abortul (este vorba de adresa logică din contorul de program PC). Registrul este READ ONLY. Poate fi șters cu INIT.

3.2.3. COMPLEMENTE PRIVIND MANAGEMENTUL MINICALCULATORULUI INDEPENDENT-102 F

3.2.3.1. **Zone de instrucțiuni și zone de date.** Avînd în vedere că la un moment dat sînt active maximum 8 pagini de memorie de cîte 4K cuvinte fiecare rezultă că un utilizator poate avea la dispoziție maximum 32K cuvinte; pentru programe mai mari sînt necesare segmentări și reîncărcarea registrelor managementului cu adresele și caracteristicile noilor pagini alocate utilizatorului respectiv. Oricum, cu ajutorul adreselor pe 16 biți pe care le prezintă procesorul sistemului de intrare-ieșire nu se poate „mătura" decît o zonă de memorie de 32K cuvinte sau 64KO (avînd în vedere că adresele cresc din 1 în 1 din octet în octet).

Pentru a se putea pune la dispoziția unui utilizator o memorie dublă (64K cuvinte), pornind de la aceeași adresă logică pe 16 biți, s-au introdus noțiunile de *zonă de instrucțiuni* (I) și *zonă de date* (D); fiecare zonă acoperă cîte o arie de memorie de maximum 32K cuvinte, caracteristicile lor fiind complet independente (cele 2 zone putînd fi disjuncte, intersectate sau chiar confundate). Intrarea într-una dintre cele 2 zone este determinată de operația care se execută:

— către zona de instrucțiuni (I) sînt direcționate toate accesele pentru citirea instrucțiunilor de executat sau pentru apelarea operanzilor care, conform modurilor de adresare, trebuie să se găsească în vecinătatea instrucțiunilor; deci, vor fi relocate în zona de instrucțiuni (I) adresele prezentate de procesor pentru:

a) citirea unei instrucțiuni (FETCH);

b) citirea unui index (care urmează întotdeauna după instrucțiune, deci în cazul modurilor de adresare 6 sau 7);

c) citirea unei adrese absolute sau apelarea unui operand imediat aflat, de asemenea, după instrucțiune (modul de adresare 2 sau 3 cu PC);

— către zona de date (D) sînt direcționate toate celelalte accese (deci cînd nu se citește o instrucțiune, un index sau un operand aflat imediat după instrucțiune).

Pentru realizarea acestui lucru s-a dublat numărul de registre de pagină activă (APR) față de managementul simplificat; în managementul completat, pe lîngă registrele de stare, există cîte 16 duble registre APR pentru fiecare mod de lucru (fig. 3-2).

Selecția unui registru APR din spațiul de instrucțiuni sau de date se face în funcție de tipul (scopul) accesului la memorie și de starea bitului corespunzător din registrul de stare suplimentar SR3 care activează zona de date (D); dacă acest bit este 0, toate accesele se fac în zona I, minicalculatoarele din această categorie (I 102F) devenind compatibile cu cele cu management simplificat; dacă bitul este 1 accesele sînt direcționate diferențiat spre cele 2 zone, conform celor arătate anterior. În funcție de adresa prezentată, pagina este aleasă conform algoritmului din fig. 3-3.

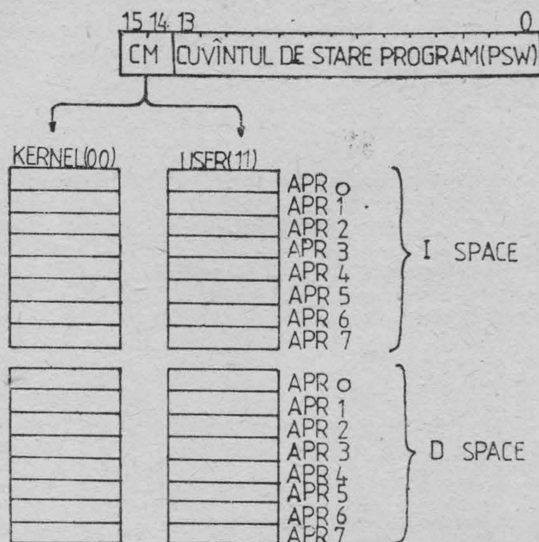


Fig. 3-2 REGISTRELE DE PAGINĂ ACTIVĂ - MGM COMPLET

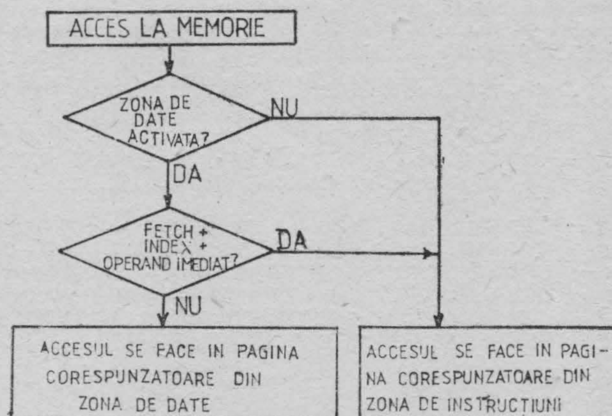


Fig. 3-3 SELECȚIA ZONEI DE INSTRUCȚIUNI SAU DE DATE

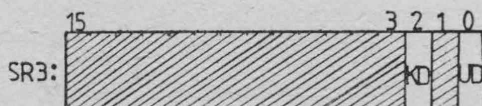
De asemenea, în funcţie de modul de lucru curent (PSW 14—15) se poate ajunge în zona de date KERNEL sau USER, respectiv în zona de instrucţiuni KERNEL sau USER. Vor exista în total 4 zone de memorie, fiecare de maximum 32K cuvinte:

KI — KERNEL INSTRUCTION
KD — KERNEL DATA
UI — USER INSTRUCTION
UD — USER DATA

3.2.3.2. **Registrele managementului completat.** O primă diferenţă faţă de registrele managementului simplificat (descris în capitolul 3.2.2.) constă în dublarea registrelor de pagină activă APR (descriptorii zonei de date fiind diferiţi faţă de cei ai zonei de instrucţiuni).

În ceea ce priveşte registrele de stare, SR0 şi SR2 sînt identice din punct de vedere logic şi funcţional cu cele descrise în capitolul precedent (vezi 3.2.2.2.); s-a adăugat un registru de stare suplimentar (SR3) folosit pentru validarea zonelor de date.

Registrul SR3 are doar 2 biţi:



p.b.0: UD = USER DATA (validare zonă de date USER);
p.b.2: KD — KERNEL DATA (validare zonă de date KERNEL).

Zonele de date sînt validate dacă biţii corespunzător din registrul SR3 sînt „1". Dacă unul dintre aceşti biţi este 0, zona corespunzătoare de date este invalidată şi toate accesele se fac în zona de instrucţiuni respectivă. Registrul SR3 este de tipul READ/WRITE, deci poate fi scris prin program validînd, respectiv invalidînd, o zonă sau ambele zone de date.

La iniţializare SR3 este şters (UD=KD=0), astfel că, neafectînd registrul, toate accesele vor fi relocate în zonele de instrucţiuni; se asigură astfel compatibilitatea cu minicalculatoarele care au management simplificat.

Avînd în vedere cele 4 zone de memorie (KI, KD, UI şi UD), precum şi faptul că fiecare dintre cele 8 pagini ale fiecărei zone este caracterizată de cîte 2 registre (descriptor de pagină — DR şi adresă pagină — AR), registrele din managementul minicalculatorului INDEPENDENT 102 F sînt:

KISDR 0—7 = KERNEL INSTRUCTION SPACE DESCRIPTOR REGISTER (registrul descriptor de pagină pentru zona de instrucţiuni KERNEL);
KISAR 0—7 = KERNEL INSTRUCTION SPACE ADRESS REGISTER (registrul adresă de pagină pentru zona de instrucţiuni KERNEL);
KDSDR 0—7 = KERNEL DATA SPACE DESCRIPTOR REGISTER
KDSAR 0—7 = KERNEL DATA SPACE ADRESS REGISTER
UISDR 0—7 = USER INSTRUCTION SPACE DESCRIPTION REGISTER
UISAR 0—7 = USER INSTRUCTION SPACE ADRESS REGISTER
UDSDR 0—7 = USER DATA SPACE DESCRIPTION REGISTER
UDSAR 0—7 = USER DATA SPACE ADRESS REGISTER

La acestea se adaugă cele trei registre de stare: SR0, SR2 şi SR3.
Adresele registrelor de stare sînt:

SR0: 777572
SR2: 777576
SR3: 772516

Adresele registrelor de pagină activă sînt date în tabelul următor.

NR. PAG.	KERNEL				USER			
	INSTRUCTION		DATA		INSTRUCTION		DATA	
	PDR (KISDR)	PAR (KISAR)	PDR (KDSDR)	PAR (KDSAR)	PDR (UISDR)	PAR (UISAR)	PDR (UDSDR)	PAR (UDSAR)
0	772300	772340	772320	772360	777600	777640	777620	777660
1	772302	772342	772322	772362	777602	777642	777622	777662
2	772304	772344	772324	772364	777604	777644	777624	777664
3	772306	772346	772326	772366	777606	777646	777626	777666
4	772310	772350	772330	772370	777610	777650	777630	777670
5	772312	772352	772332	772372	777612	777652	777632	777672
6	772314	772354	772334	772374	777614	777654	777634	777674
7	772316	772356	772336	772376	777616	777656	777636	777676

TABEL. ADRESELE REGISTRELOR DE PAGINĂ ACTIVĂ — MGM COMPLET

3.2.4. INSTRUCȚIUNI DE COMUNICARE ÎNTRE MODUL DE LUCRU CURENT ȘI ANTERIOR

Instrucțiunile prezentate în continuare asigură comunicarea între două spații (de instrucțiuni sau de date), unul corespunzător modului curent și altul modului de lucru anterior.

2.3.4.1. Instrucțiunile MFPI și MFPD

— *mnemonică și cod:*

MFPI	0065SS
MFPD	1065SS

— *nume:* move from previous instruction/data space (transfer din spațiul anterior de instrucțiuni, respectiv date).

— *operație:* **(SP) ← (src);** un cuvînt de la adresa sursă calculată de management conform modului de lucru anterior (PSW 12—13) este depus în stiva modului curent (dat de PSW 14—15); operandul sursă va fi luat din zona de instrucțiuni (MFPI), respectiv de date (MFPD), cele două zone diferind dacă zona de date există și este validată.

— *poziționarea indicatorilor de condiții:*
N se poziționează conform semnului operandului sursă;
Z=1 dacă sursa transferată este nulă; în rest, Z=0;
V=0;
C este neafectat.

2.3.4.2. Instrucțiunile MTPI și MTPD

— *mnemonică și cod:*

MTPI	0066DD
MTPD	1066DD

— *nume:* move to previous instruction/data space (transfer în spațiul anterior de instrucțiuni, respectiv date).

— *operație:* **(dst)←(SP)↑;** un cuvînt din stiva modului curent este depus la destinația calculată de management conform modului de lucru anterior (PSW 12—13). Destinația va fi în zona de instrucțiuni (MTPI) sau de date (MTPD), cele 2 zone diferind dacă zona de date există și este validată.

— *poziţionarea indicatorilor de condiţii:*
N va marca semnul operandului transferat;
Z=1 dacă operandul transferat este nul; în rest, Z=0;
V=0;
C este neafectat.

Observaţii: a) toate aceste instrucţiuni utilizează stiva modului de lucru curent (dat de PSW 14—15); MFPI şi MFPD execută o depunere pe stivă (push), iar MTPI şi MTPD execută o preluare de pe stivă (pop); pointerul stivei (SP) va fi prelucrat corespunzător;

b) pentru ca relocarea adresei operandului să se execute şi să ţină cont de modul de lucru anterior (din PSW 12—13), precum şi de zona de memorie apelată (instrucţiuni sau date), trebuie ca managementul să fie ON;

c) în cazul minicalculatoarelor cu management simplificat (pentru care zona de instrucţiuni coincide cu cea de date), instrucţiunile MFPD şi MTPD nu sînt considerate rezervate sau ilegale, ele fiind identice în execuţie cu MFPI şi respectiv MTPI.

3.2.5. PROGRAMARE

3.2.5.1. Procedura de relocare a adreselor

Procesorul poate prezenta adrese de bus pe 16 biţi (numite *adrese logice);* acestea acoperă intervalul de adrese pe cuvînt 000 000 — 177776, deci un domeniu de memorie de 32K cuvinte. Pentru minicalculatoarele fără management sau cu management dezactivat (OFF) adresa logică coincide cu cea *fizică* (de adresare a memoriei).

Dacă managementul există şi este activat (ON), are loc procesul de relocare a adresei logice, obţinîndu-se o adresă fizică pe 18 (sau mai mulţi) biţi, adresă capabilă să adreseze o memorie de capacitate mai mare (54K cuvinte, 128K cuvinte, etc.). În principiu, relocarea constă în a aduna adresa logică cu o bază de adresare numită *constantă de relocare.*

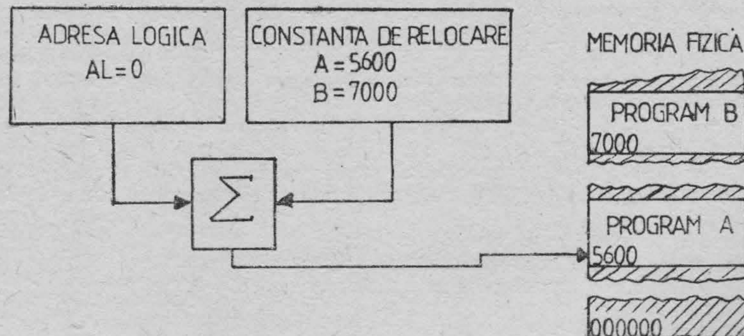


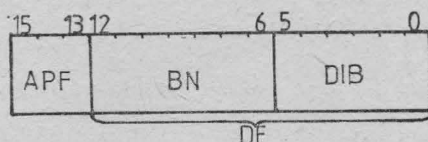
Fig. 3-4 PRINCIPIUL DE RELOCARE A ADRESELOR

Programul B este relocat de la adresa 70 000 (nu 7 000 ca în fig. 3—4)

În figura 3—4. este prezentat principiul de relocare. Presupunem existenţa în memorie a două programe, programul A relocat de la adresa 5600(8) şi programul B relocat de la adresa 70000(8). Fiecare program începe la adresa logică 0. Cînd este lansat programul A, adresa 0 este adunată cu constanta de relocare 5600, rezultînd adresa fizică 5600; dacă următoarea adresă logică din programul A este 2, ea va fi relocată la adresa fizică 5602. Cînd este rulat programul B, constanta de relocare este schimbată cu 70000 şi procesul de adresare decurge analog. Utilizînd constantele de relocare (aflate în registrele de pagină activă), nu mai este nevoie să se recal-

culeze adresele fizice ale unui program ori de cît ori este încărcat într-o altă zonă de memorie; din punctul de vedere al programatorului rutina începe la aceeași adresă logică, managementului revenindu-i rolul de a reloca adresele conform disponibilităților din memoria fizică.

Pentru a genera adresa fizică, managementul interpretează adresa logică prezentată de procesor în modul următor:



Adresa fizică este împărțită în două cîmpuri principale:

1. APF=ACTIVE PAGE FIELD (cîmpul de pagină activă): acești 3 biți (pozițiile 13—15, cele mai semnificative) determină care pagină (dintre cele 8 active la un moment dat) va fi utilizată pentru relocarea adresei, selectînd registrele APR corespunzătoare; în acest fel se cunoaște constanta de relocare (practic adresa de început a paginii), precum și caracteristicile paginii (lungimea, direcția de dezvoltare, cheia de acces; vezi configurația registrelor APR, paragraful 3.2.2.1.).

2. DF=DISPLACEMENT FIELD (cîmpul deplasament); cei 13 biți mai puțin semnificativi ai adresei logice conțin adresa relativă din pagină; cei 13 biți limitează lungimea unei pagini la maximum 4 K cuvinte sau 8 KO(2^{13}=8 KO). Acest cîmp este la rîndul său subdivizat în două părți:

2.1. BN=BLOCK NUMBER (numărul blocului); cei 7 biți din pozițiile 6—12 ale adresei logice sînt interpretați ca număr de bloc din pagina respectivă.

2.2. DIB=DISPLACEMENT IN BLOCK (deplasamentul în bloc): cei 6 biți mai puțin semnificativi (pozițiile 0—5) conțin adresa relativă din cadrul unui bloc.

Calculul adresei fizice este prezentat în figura 3-5.

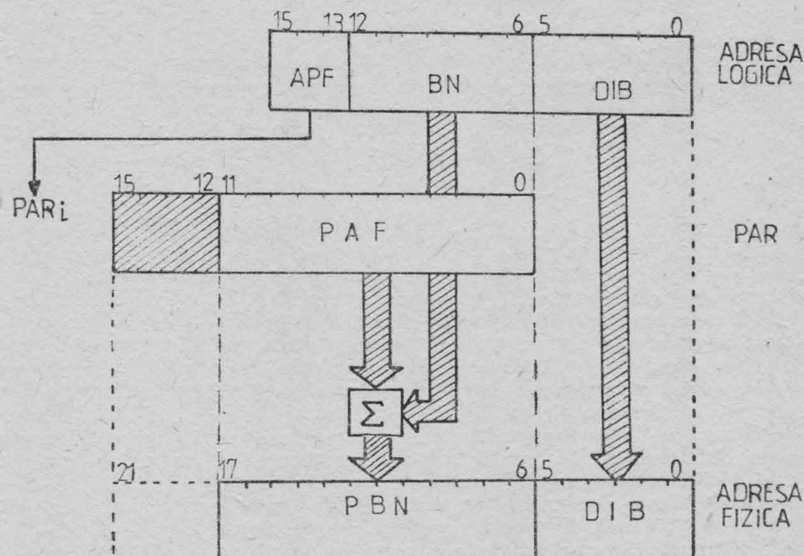


Fig.3-5 CALCULUL ADRESEI FIZICE

Secvenţa logică urmată pentru a construi adresa fizică este următoarea:

1. Este selectat un set de registre APR (registre de pagini active) în funcţie de modul de lucru curent indicat în PSW 14—15 (KERNEL sau USER); în cazul managementului cu zone de date activate, selecţia va ţine cont şi de tipul accesului la memorie (zonă de instrucţiuni sau zonă de date).

2. Cu cîmpul APF (poziţiile 13—15) din adresa logică se selectează o pagină (adică o pereche de registre APR corespunzătoare unei pagini) dintre cele 8 pagini active la un moment dat (APR0-APR7).

3. Cîmpul PAF (PAGE ADRESS FIELD) din registrul PAR selectat (APR= =PAR+PDR) conţine adresa de început a paginii în memoria fizică (aceasta este de fapt un număr de bloc din memorie, nefiind luaţi în considerare cei 6 biţi mai puţin semnificativi); acest cîmp este adunat cu numărul blocului din adresa logică (BN). rezultînd numărul blocului fizic din memorie (PBN=PHISICAL BLOCK NUMBER).

4. Deplasamentul în cadrul blocului (DIB) rămîne nemodificat, constituind biţii cei mai puţin semnificativi (0—5) din adresa fizică.

Constanta de relocare este deci cantitatea PAF din registrul PAR selectat; este clar că schimbînd această constantă, adresa logică va fi plasată (relocată) în altă zonă (pagină) de memorie. Se observă că adresa fizică rezultă pe 18 biţi; într-o ulterioară extensie adresa poate ajunge pînă la maximum 22 biţi, utilizînd întreaga capacitate a registrelor PAR.

Cei trei biţi mai semnificativi ai adresei logice determină perechea de registre PAR-PDR utilizată pentru relocare. În tabelul următor este prezentată dependenţa între adresa logică şi perechea de registre PAR-PDR selectată:

INTERVALUL DE ADRESE LOGICE	PERECHEA PAR-PDR SELECTATĂ
000 000 — 017776	0
020 000 — 037776	1
040 000 — 057776	2
060 000 — 077776	3
100 000 — 117776	4
120 000 — 137776	5
140 000 — 157776	6
160 000 — 177776	7

Rezultă că lungimea maximă a unei pagini este de 4096 cuvinte (4 Kcuvinte), adresele relative într-o pagină acoperind domeniul de adrese:

00000 — 17776.

Utilizînd toate cele 8 perechi PAR — PDR ce pot fi active la un moment dat într-un mod de lucru, rezultă că un program (nesegmentat) poate avea o lungime de maximum 8 pagini, adică 32 768 de cuvinte (32 Kcuvinte sau 64 Kocteţi).

Fiecare pagină este relocată independent, registrul PAR corespunzător putînd fi încărcat cu orice constantă de relocare. Rezultă de aici:

— două pagini logic consecutive pot acoperi două spaţii oarecare din memoria fizică (nu neapărat adiacente); de asemenea, o pagină cu număr logic mic poate fi relocată la o adresă fizică mare şi invers, neexistînd o relaţie între rangul adreselor logice şi cel al adreselor fizice;

— două sau mai multe pagini pot fi relocate în acelaşi spaţiu de memorie fizică; în acest fel, o rutină încărcată într-o anumită zonă de memorie poate fi utilizată de mai multe programe.

În figura 3-6. este dat un exemplu de relocare a întregului domeniu de adrese logice (000 000 — 177776) în diverse zone ale memoriei fizice.

Studiind figura 3-6. se pot deduce regulile de aflare a adresei fizice cînd se cunoaşte adresa logică şi conţinutul PAR-ului corespunzător sau, invers, cum se poate determina constanta de relocare pornind de la adresa logică şi cunoscînd adresa fizică la care se doreşte să se ajungă.

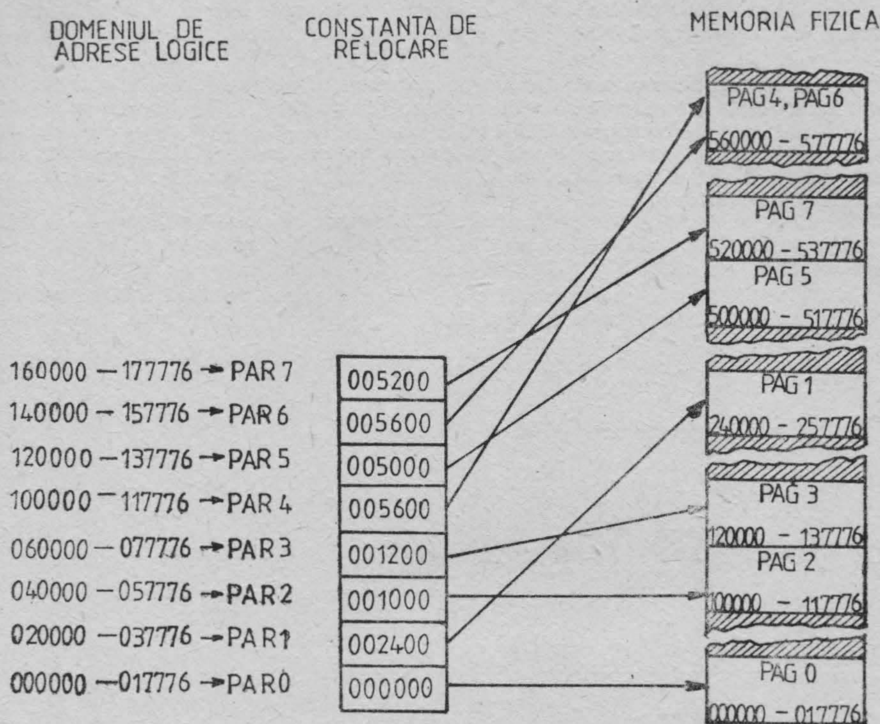


Fig. 3-6 DEFINIREA PAGINILOR ÎN MEMORIE

De exemplu, pentru determinarea adresei fizice se procedează astfel:

— se examinează biții 12—15 ai adresei logice, precum și modul curent de lucru (PSW 14—15), pentru a afla care registru PAR conține constanta de relocare corespunzătoare;

— se citește registrul PAR respectiv și se deplasează conținutul său spre stînga cu 2 cifre octale (adăugîndu-se practic 2 de 0 în dreapta constantei de relocare);

— se adună constanta de relocare astfel prelucrată cu conținutul pozițiilor 0—12 al adresei logice; suma rezultată reprezintă adresa fizică (pe 18 biți) căutată.

Exemple: 1) Ce adresă fizică îi corespunde adresei logice 123456 știind că PAR5 conține 006426? Se consideră modul de lucru KERNEL. Toate numerele sînt octale.

Biții cei mai semnificativi ai adresei logice selectează PAR5 din setul KERNEL (registru ce are adresa 772352); constanta de relocare cadrată (deplasată spre stînga) este 642600, deci rezultă adresa fizică:

$$AF = 642600 + 003456 = 646256.$$

2) Ce registru PAR trebuie încărcat și cu ce constantă de relocare pentru a plasa adresa logică 037654 la adresa fizică 472354? (se știe că ultimele două cifre octale ale adreselor logică și fizică — deplasamentul în bloc — coincid). Se consideră modul de lucru KERNEL.

Descompunînd adresa logică 037654 obținem pagina (perechea PAR-PDR) 1 și deplasamentul 17654. Executăm diferența:

$$472354—17654 = 452500;$$

renunțînd la ultimele 2 cifre octale obținem constanta de relocare 4525. Rezultă că trebuie încărcat registrul PAR1 din setul KERNEL (adresa 772342) cu cantitatea: 004525.

3.2.5.2. **Reprezentarea unei pagini în memorie.** După cum s-a văzut din capitolul 3.2.2.1. unde a fost prezentat descriptorul de pagină (PDR), lungimea unei pagini este indicată de constanta PLF (PDR pozițiile 8—14). PLF reprezintă numărul de blocuri adiacente pe care le conține pagina minus 1 (deci PLF=0 marchează o pagină cu un bloc); fiecare bloc are o lungime fixă, de 32 cuvinte. O pagină poate avea între 1 și 128(10) blocuri.

Totodată, în funcție de starea bitului ED, pagina se poate dezvolta de la adrese mici spre adrese mari (ED=0, UPWARD EXPANSION) sau de la adrese mari spre adrese mici (ED=1, DOWNWARD EXPANSION).

În figura 3-7. este reprezentată o pagină cu adresa de bază 026000, cu 122(8) blocuri și cu ED=0; registrele PAR și PDR asociate paginii vor avea conținuturile: PAR=000260 și PDR=050406, deoarece:
— considerăm cheia de acces ACF=3 (read/write);
— PLF=122—1=121(8).

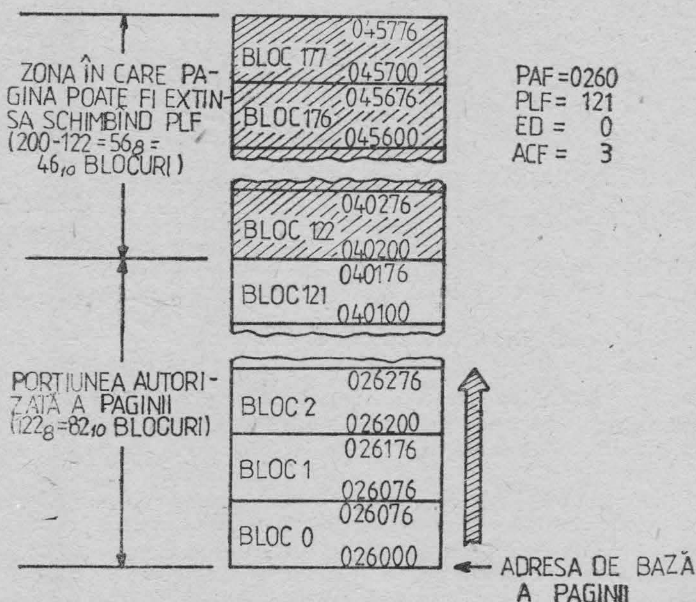


Fig.3-7 PAGINĂ DEZVOLTATĂ SPRE ADRESE MARI

În exemplul prezentat, porțiunea autorizată a paginii este cuprinsă între adresele 026000 și 040176 (considerînd o organizare pe cuvînt — adrese pare). Orice încercare de acces în zona neatorizată a paginii va provoca un abort depășire lungime pagină (deci pentru orice adresă logică în care numărul logic al blocului este mai mare decît 121(8), sau pentru orice adresă fizică rezultată mai mare sau egală cu 040200).

În figura 3-8. este reprezentată o altă pagină cu aceeași adresă de bază (026000), cu același număr de blocuri (122(8)), dar cu ED=1 (DOWNWARD EXPANSION); pagina va porni de la adresa sa maximă posibilă (045776) și va evolua spre adrese descrescătoare. Registrele descriptoare PAR și PDR vor avea conținuturile:

PAR=000260 și PDR=050416.

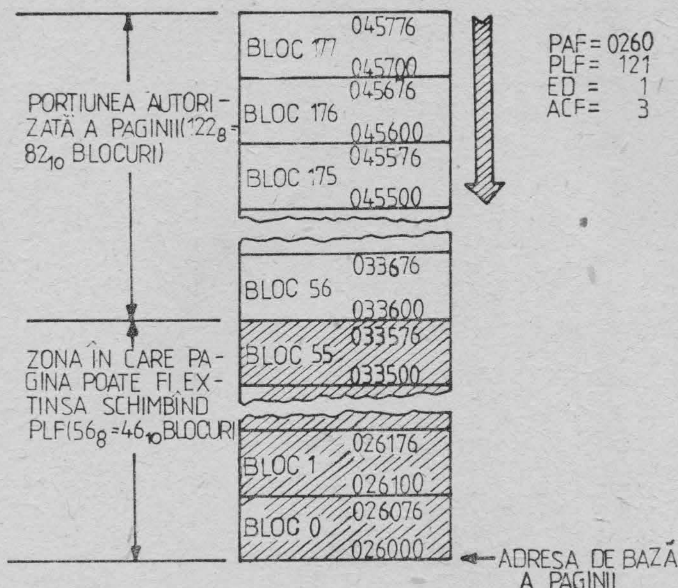


Fig.3-8 PAGINĂ DEZVOLTATĂ SPRE ADRESE MICI

De data aceasta, porțiunea autorizată a paginii este cuprinsă între adresele 033600 și 045776. Abortul depășire lungime pagină apare în cazul prezentării unei adrese fizice mai mici decît 033600 (sau cu numărul logic al blocului mai mic sau egal cu 55(8)).

Se pot da următoarele relații de calcul ale limitelor zonei autorizate din cadrul unei pagini cunoscînd adresa de bază a paginii (AB), precum și constanta PLF care determină lungimea paginii în număr de blocuri:

1° ED=0 (UPWARD EXPANSION):
— limita inferioară: AB
— limita superioară: AB+76+PLF×100
2° ED=1 (DOWN WARD EXPANSION):
— limita inferioară: AB+17700—PLF×100
— limita superioară: AB+17776.

S-a considerat organizarea memoriei pe cuvinte (adrese pare); toate calculele se execută în octal.

3.2.5.3. **Testarea unei pagini de memorie.** Cu programul prezentat în continuare se poate testa (sumar) o pagină oarecare din memoria centrală a minicalculatorului. Testul constă în:

1) scrierea întregii pagini cu o cantitate oarecare (pe cuvînt);

2) verificarea scrierii efectuată anterior, prin comparare; dacă rezultatul verificării este corect, programul se buclează (verificare la infinit), putîndu-se astfel detecta eventuale pierderi în timp ale informației; în caz de eroare programul se oprește în HALT.

Pentru alegerea paginii de testat este util să se cunoască limitele între care se pot întinde paginile din memoria centrală, considerînd că prima pagină începe la adresa 000 000 și că toate paginile au lungimea maximă (4 K cuvinte). În tabelul din fig. 3-9. sînt prezentate paginile de lungime maximă în care poate fi împărțită memoria, limitele acestora, precum și constanta de relocare PAF ce trebuie încărcată în registrul PAR corespunzător pentru ca adresa logică relativă 0 să corespundă, în urma relocării, cu adresa de bază a paginii.

NR. PAG.	LIMITELE FIZICE ALE PAGINI	PAF
0	000000 — 017776	0000
1	020000 — 037776	0200
2	040000 — 057776	0400
3	060000 — 077776	0600
4	100000 — 117776	1000
5	120000 — 137776	1200
6	140000 — 157776	1400
7	160000 — 177776	1600
8	200000 — 217776	2000
9	220000 — 237776	2200
10	240000 — 257776	2400
11	260000 — 277776	2600
12	300000 — 317776	3000
13	320000 — 337776	3200
14	340000 — 357776	3400
15	360000 — 377776	3600
16	400000 — 417776	4000
17	420000 — 437776	4200
18	440000 — 457776	4400
19	460000 — 477776	4600
20	500000 — 517776	5000
21	520000 — 537776	5200
22	540000 — 557776	5400
23	560000 — 577776	5600
24	600000 — 617776	6000
25	620000 — 637776	6200
26	640000 — 657776	6400
27	660000 — 677776	6600
28	700000 — 717776	7000
29	720000 — 737776	7200
30	740000 — 757776	7400
31	760000 — 777776	7600

Fig. 3-9. Împărţirea memoriei în pagini.

Observaţii:

— paginile 0—6 (deci pînă la adresa maximă 157776) pot fi accesate şi cu adrese logice (managementul dezactivat); de la adresa 160000 în sus este necesar ca managementul să fie activat, iar registrul PAR asociat să conţină constanta corespunzătoare de relocare;

— pagina 31 (ultima) nu aparţine memoriei centrale ci adreselor de serviciu (vezi cap. 1.2.2.); această ultimă pagină nu poate fi accesată de către procesor, chiar dacă ea există fizic în modulele de memorie.

Programul propus în continuare este scris în pagina 0 şi va putea testa orice altă pagină (1—30); se presupune, deci, că pagina 0 este bună. Pentru a putea testa orice pagină din memorie trebuie activat managementul; din momentul activării toate adresele vor fi relocate, inclusiv cele ale programului; rezultă că vor trebui definite 2 pagini: pagina 0, în care se află programul şi o a doua pagină, cea de testat. Pentru a se determina uşor depăşirea adresei logice (sfîrşitul paginii) s-a ales pentru pagina de testat setul PAR3-PDR3, deci adresele logice vor evolua în intervalul:

060 000 — 077 776 .

Faptul că adresa logică ajunge la valoarea 100 000 (prima adresă de cuvînt ce urmează după 077 776) se determină uşor folosind instrucţiunea TST (bitul 15 — de semn — este 1 şi adresa va fi interpretată de instrucţiune ca un număr negativ).

Programul folosește stiva pentru eventualele erori ce pot apărea, fiecăreia corespunzîndu-i o întrerupere internă:

```
TIME — OUT (adresă inexistentă)      cu vectorul 004;
EROARE PARITATE                      cu vectorul 114;
EROARE MANAGEMENT                    cu vectorul 250.
```

Programul este următorul:

```
004     PCIT                        TRATARE ÎNTRERUPERI
006     000000
114     PCIT
116     000000
250     PCIT
252     000000
START   MOV #START, SP              INIȚIALIZARE STIVĂ
        MOV #DATA, R1               DATA→R1
        MOV #PAF, @# PAR3           DEFINIRE PAGINĂ LOGICĂ 3
        MOV #077406, @# PDR3
        CLR @ #PAR0                 DEFINIRE PAGINĂ LOGICĂ 0
        MOV #077402, @ # PDR0
        INC @ #CPSR                 ACTIVARE CONTROLOR PARITATE
        INC @ #SRO                  ACTIVARE MANAGEMENT (MGM ON)
        MOV #060000, R0             SCRIERE PAGINĂ CU DATA (R1)
A1      MOV R1, (R0)+
        TST R0
        BPL A1
A4      MOV #060000, R0             VERIFICARE PAGINĂ
A3      CMP R1, (R0)+
        BNE A2
        TST R0
        BPL A3
        BR  A4
A2      HALT                        EROARE VERIFICARE
PCIT    TST —(R0)                   DECREMENTARE R0 cu 2
        RTI                         ÎNTOARCERE DIN ÎNTRERUPERE
```

Observații:

— se consideră modul de lucru KERNEL;

— definirea constantelor utilizate:

PCIT =adresa rutinei de tratare a întreruperilor (de exemplu, poate fi 2000);

START=adresa de lansare a programului (de exemplu: 1000); programul evoluează spre adrese crescătoare, iar stiva va evolua (începînd de la 776) spre adrese descrescătoare;

DATA =configurația de test (un cuvînt); se poate modifica oricînd (are adresa START+6);

PAF =constanta de relocare corespunzătoare a paginii ce se dorește testată (vezi tabelul din fig. 3-9); modificînd PAF se pot testa diferite pagini de memorie (numai paginile 1—30);

PAR3 =172346;

PDR3 =172306;

PAR0 =172340;

PDR0 =172300;

CPSR =adresa registrului de stare al controlorului de paritate ce poate fi cuprinsă între 172100 și 172136, în funcție de tipul minicalculatorului și de planul de memorie pe care se află pagina de testat (vezi observația de la sfîrșitul paragrafului 3.1.2.);

SR0 =177572 (registrul de stare 0 al managementului);

— primele 6 instrucțiuni ale programului inițializează stiva (necesară în cazul întreruperilor) și cele două pagini utilizate: pagina logică 0 pentru program și pagina logică 3 care va fi testată; programul trebuie să existe în totalitate în pagina 0, deci între adresele 000000 și 017776; cheia de acces a paginii 0 a fost definită 1 deci această pagină este protejată la scriere;

— instrucţiunea a 7-a activează controlorul de paritate asociat paginii testate; dacă nu se doreşte test de paritae se pot înlocui cele două cuvinte ale instrucţiunii cu două NOP-uri (cod: 000240);

— instrucţiunea a 8-a activează managementul de memorie, deci toate adresările ulterioare vor fi relocate; se aprinde indicatorul luminos MGM ON sau RELOC pe panoul frontal;

— instrucţiunile 9—12 realizează scrierea întregii pagini cu o cantitate oarecare încărcată în prealabil în registrul R1;

— ultima parte a programului verifică pagina scrisă anterior pînă la apariţia unei eventuale necoincidenţe între conţinutul lui R1 şi o locaţie de memorie, caz în care programul se opreşte în HALT-ul de la adresa A2; R0 (afişat la panou) conţine adresa logică a cuvîntului eronat +2;

— în caz de eroare hardware (time-out, eroare paritate sau eroare management) se ajunge în rutina PCIT de tratare a întreruperii corespunzătoare; programul se reia din punctul în care a fost întrerupt dar după ce se decrementează adresa de memorie (R0), obţinîndu-se astfel o buclare cu eroare pe aceeaşi adresă. Pentru a determina eroarea hardware ce apare se pot separa cele 3 întreruperi, tratîndu-le la 3 adrese distincte (PCIT 1, PCIT 2 şi PCIT 3); în subprogramele de tratare se vor plasa HALT-uri la adrese cunoscute;

— pentru buclarea unei operaţii pe o singură adresă se poate suprima incrementarea adresei de memorie din registrul R0 prin înlocuirea modului de adresare 2 (autoincrement) cu modul de adresare 1; de exemplu, la adresa A3, instrucţiunea CMP R1, (R0)+devine CMP R1, (R0); de asemenea, testul conţinutului lui R0 se va înlocui cu NOP-uri;

— operaţiunile pot fi executate şi pe byte, înlocuind instrucţiunile de la adresele A1 şi A3 cu instrucţiuni pe byte.

3.3. Memoria cache

3.3.1. ROLUL ŞI LOCUL MEMORIEI CACHE ÎN SISTEM

Memoria cache este o memorie de capacitate mică dar de viteză ridicată, care menţine la dispoziţia procesorului o copie a unei porţiuni din memoria centrală selectată anterior de acesta, permiţînd un nou acces rapid la instrucţiuni şi date. Memoria cache este transparentă software, utilizarea ei neimpunînd nici un fel de modificare a programelor. Pentru utilizator prezenţa memoriei cache se face simţită numai prin creşterea vitezei de lucru a procesorului. Cu ajutorul acestei extensii se obţine o îmbunătăţire a performanţelor calculatorului cu preţ de cost suplimentar scăzut.

Locul memoriei cache în sistem este între procesor (incluzînd şi managementul) şi busul cu memoria centrală a calculatorului, după cum rezultă din figura 3-10.

Memoria cache este accesibilă numai procesorului; acesta „caută" datele de care are nevoie în cache şi, dacă le găseşte, renunţă la accesul lung pe care ar fi trebuit să-l facă spre memoria centrală. Avînd în vedere că un ciclu de memorie cache durează mult mai puţin decît un ciclu al memoriei centrale, rezultă o creştere sensibilă a vitezei de lucru a procesorului. De fapt, eficienţa memoriei cache decurge din observaţii statistice, făcute asupra programelor ce se rulează în mod normal pe un calculator; s-a constatat că pentru execuţia unei secvenţe normale de program, procesorul lansează mult mai multe operaţii de citire din memorie decît operaţii de scriere; totodată, majoritatea rutinelor implică execuţia de mai multe ori a aceloraşi instrucţiuni situate într-o zonă restrînsă de memorie (bucle de program).

Din experimentele efectuate s-a constatat că eficienţa memoriei cache este maximă dacă procesorul este dotat şi cu posibilitatea de execuţie cu suprapunere de faze (de exemplu, citirea instrucţiunii următoare în timp ce se execută instrucţiunea citită anterior). Procesorul minicalculatorului **INDEPENDENT 102 F** foloseşte asociaţia dintre cele două concepte (cache şi suprapunere de faze) realizînd o viteză ridicată de execuţie a instrucţiunilor dintr-un ciclu tipic de program.

Există mai multe tipuri de memorii cache. Cea implementată pe minicalculatorul **I-102 F** şi pe variantele dezvoltate ale familiei **CORAL** are următoarele caracteristici:

— *reactualizare automată şi în permanenţă*, în funcţie de accesele pe care le face procesorul spre memorie;

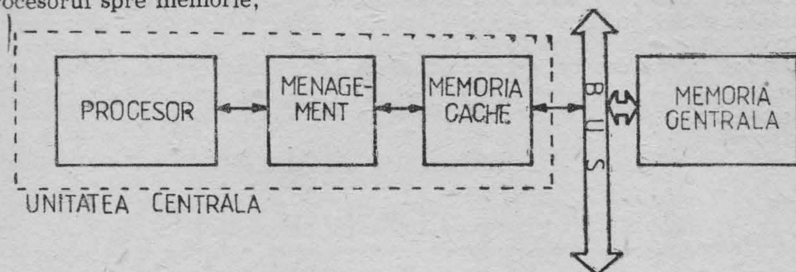


Fig. 3-10 LOCUL MEMORIEI CACHE IN SISTEM

— *mecanism de adresare prin mapare directă:* fiecare cuvînt din memoria centrală poate fi adus într-un anumit loc, unic, din cache; la un moment dat, o singură adresă poate fi recunoscută, prin comparare;

— dacă un cuvînt citit din memoria centrală nu se află în cache, el este scris în cache în *unica* locaţie ce-i este rezervată;

— *scriere prin transparenţă:* data scrisă în cache este scrisă imediat şi în memoria pricipală (aceasta conţinînd în orice moment o copie validă a tuturor datelor);

— *capacitate:* 1K cuvînt (2KO=2 048 octeţi).

După cum se va vedea mai departe, memoria cache conţine la un moment dat o copie a adreselor modulo 1K din memoria centrală accesate pînă în acel moment.

3.3.2. ORGANIZAREA DATELOR ÎN CACHE

În figura 3-11 este prezentat modul de organizare a datelor în memoria cache.

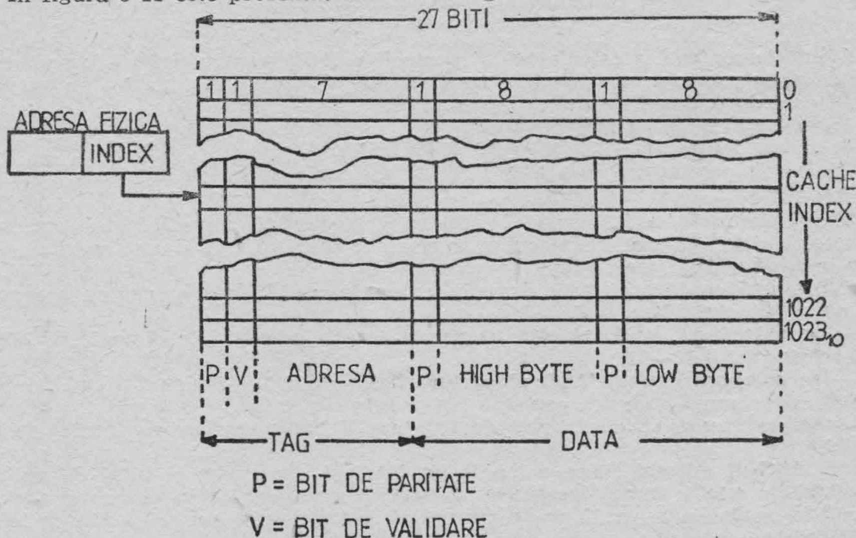


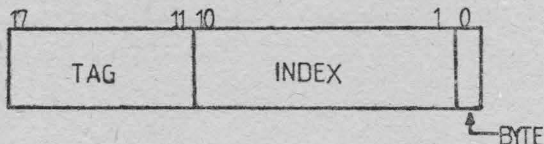
Fig. 3-11 ORGANIZAREA MEMORIEI CACHE

Cei 2 048 bytes din memoria centrală sînt organizați în 1 024 cuvinte de cîte 27 de biți. Fiecare cuvînt din cache este alcătuit din două cîmpuri:

1) **cîmpul TAG**, care conține 7 biți de adresă, un bit de validare (V) și bitul de paritate (P) al întregului cîmp; biții de adresă coincid cu cei 7 biți mai semnificativi ai adresei fizice de memorie la care se află DATA corespunzătoare (reprezintă, deci, numărul zonei de memorie de 1K cuvînt dintre cele 128 de zone ale întregii memorii);

2) **cîmpul DATA** conține cei doi octeți de date, fiecare cu bitul său de paritate.

Pentru adresarea memoriei cache, adresa fizică (pe 18 biți) este împărțită în următoarele cîmpuri:



Cîmpul BYTE selectează unul dintre cei doi octeți ai cuvîntului:

BYTE=0: este selectat octetul inferior (LOW), deci cel de la adresa pară;

BYTE=1: este selectat octetul superior (HIGH), deci cel de la adresa impară.

Cîmpul INDEX determină care poziție din cache este utilizată pentru a păstra copia datei din memoria centrală; acest cîmp va adresa unul dintre cele 1 024 de cuvinte de cîte 27 de biți din memoria cache.

Cîmpul TAG reprezintă numărul blocului de memorie (dintre cele 128 blocuri, considerînd capacitatea maximă a memoriei) în care se află data ce este copiată în cache. Constanta TAG este memorată în locația corespunzătoare din cache o dată cu memorarea datei respective.

Se observă că TAG și INDEX reconstituie întreaga adresă pe 18 biți a unui cuvînt de memorie; copia din memoria centrală poate fi astfel identificată în cache fără pericolul unei adresări ambigue.

3.3.3. CICLURI HIT ȘI MISS

Memoria centrală a calculatorului poate fi referită pentru cicluri de scriere sau citire de către procesor sau de către un dispozitiv de tip DMA conectat pe bus; vom deosebi deci două tipuri de accese la memorie:

1) *accese din partea procesorului;*

2) *accese* NPR (NON PROCESOR REQUEST).

Vom vedea în continuare cum este afectată memoria cache de un acces anumit la memoria centrală a calculatorului.

3.3.3.1. **Accesele la memorie din partea procesorului.** Adresa logică prezentată de procesor este relocată de blocul management (dacă există și este validat), obținîndu-se adresa fizică (pe 18 biți).

Procesorul caută mai întîi data în memoria rapidă cache; pentru aceasta se parcurg următoarele faze:

— se selectează cuvîntul din memoria cache adresat de cîmpul INDEX al adresei fizice (biții 10—01);

— se compară cîmpul TAG al adresei fizice (biții 11—17) cu TAG-ul memorat în locația corespunzătoare din cache;

— se cercetează starea bitului de validare V corespunzător cuvîntului selectat din cache.

Dacă TAG-ul adresei fizice coincide cu TAG-ul din cache și bitul de validare corespunzător este 1 atunci înseamnă că data căutată se află în cache (sau locația adresată în memoria centrală are copie validă în cache); se spune că a avut loc un **HIT** (potrivire).

În caz de nepotrivire, ciclul se numește **MISS**.

În funcție de operația executată (scriere sau lectură) și de tipul ciclului cache (HIT sau MISS) avem următoarele cazuri:

1) *Lectură cu* HIT; datele căutate se află în cache, deci nu se mai cere busul pentru acces la memoria centrală, operațiunea reducîndu-se la un ciclu rapid de cache. Evident, conținuturile cache-ului și memoriei centrale rămîn nemodificate.

2) *Lectură cu* MISS; datele căutate nu se află în cache, deci operația se va executa în memoria centrală. În cache are loc operația de ALOCARE: cuvîntul din cache de la adresa dată de INDEX este încărcat cu data citită din memoria centrală și cu cîmpul TAG corespunzător din adresa fizică; de asemenea, bitul de validare V ia valoarea 1 (validînd datele alocate).

3) *Scriere cu* HIT; locația în care urmează să se execute scrierea a fost anterior alocată. Scrierea se va executa atît în cache (în locația corespunzătoare) cît și în memoria centrală; această operație se numește ACTUALIZARE (TAG-ul rămîne neschimbat; de asemenea, V se menține în 1).

4) *Scriere cu* MISS; locația în care urmează să se execute scrierea nu a fost anterior alocată. Scrierea se va executa în memoria centrală, iar în cache va avea loc operația de ALOCARE (analog cu cazul 2); se asigură astfel existența în cache a unei copii valide a memoriei centrale.

În cazul operațiilor pe byte lucrurile decurg analog, cu excepția operației scriere cu MISS (conținutul cache-ului rămîne nemodificat).

Se observă că eficiența cache-ului apare numai în cazul unei operații de LECTURĂ cu HIT (singurul caz în care este suprimat accesul la memoria centrală). Cercetările statistice arată, însă, că, într-un program tipic, au loc aproximativ 90% lecturi, față de numai 10% scrieri; dintre aceste lecturi, ciclurile HIT apar cu o frecvență de 77% pînă la 92%. Rezultă că, în mod normal, se poate conta pe un procent de 70—80% de accese rapide în memoria cache față de totalul acceselor solicitate de procesor spre memoria sistemului. Se apreciază că, în cazul minicalculatorului I-102 F, eficiența memoriei cache se exprimă printr-o creștere a vitezei de calcul a procesorului de la 2 la 2,5 ori.

3.3.3.2. **Accese la memorie de tip NPR.** Din punctul de vedere al cache-ului există probleme numai în cazul în care un periferic (DMA) modifică (scrie) conținutul unuia sau mai multor cuvinte din memoria centrală ce sînt alocate în cache; în urma scrierii, copia din cache nu va mai corespunde cu memoria centrală și data respectivă va trebui să fie invalidată. Deci, în cazul unei scriei cu HIT (TAG-ul adresei fizice corespunde cu TAG-ul memorat în cache și bitul de validare corespunzător este 1) va avea loc invalidarea cuvîntului din cache de la adresa dată de INDEX prin ștergerea bitului V (V=0).

În cazul unei lecturi sau a unei scrieri cu MISS evident că nu apare nici o modificare în cache.

În tabelul din figura 3-12, sînt prezentate toate operațiile ce pot apărea pe busul memoriei, precum și efectul lor asupra memoriei cache.

PROCESOR

OPERAȚIA	CE SE ÎNTÎMPLĂ ÎN CACHE	CE SE ÎNTÎMPLĂ ÎN MEMORIA CENTRALĂ
Lectură (cuvînt, byte)		
HIT	neafectat	neafectat
MISS	alocare	neafectat
Scriere (cuvînt)		
HIT	actualizare	scriere
MISS	alocare	scriere
Scriere (byte)		
HIT	actualizare	scriere
MISS	neafectat	scriere

NPR

OPERAŢIA	CE SE ÎNTÎMPLĂ IN CACHE	CE SE ÎNTÎMPLĂ ÎN MEMORIA CENTRALĂ
Lectură		
HIT	neafectat	neafectat
MISS	neafectat	neafectat
Scriere (cuvînt, byte)		
HIT	invalidare	scriere
MISS	neafectat	scriere

Fig. 3-12. OPERAŢIILE CU MEMORIA CACHE.

3.3.4. REGISTRELE MEMORIEI CACHE

Ca oricărui dispozitiv al sistemului şi memoriei cache i s-au asociat registre de serviciu; acestea au scopul de a facilita testarea extensiei, precum şi de a semnaliza eventualele erori de paritate ce pot apărea în timpul operaţiilor de lectură. Structura registrelor ,precum şi utilizarea lor, depinde de modul în care a fost implementată memoria cache. Există diferenţe de la un tip de calculator la altul. În general, aceste registre pot fi citite de utilizator (avînd asociate adrese de serviciu), dar nu pot fi scrise decît în condiţii speciale de testare a memoriei cache (mod maintenance). De regulă testarea memoriei cache se face prin microprograme de diagnoză, deci prin proceduri hardware.

În principiu, memoria cache trebuie să deţină două registre importante şi anume un registru de erori şi un registru de control.

a) *Registrul de erori al memoriei cache*
Adresa: 777744



p.b.5: TAG PARITY ERROR (eroare paritate în cîmpul TAG);
p.b.6: LOW BYTE PARITY ERROR (eroare paritate octet inferior);
p.b.7: HIGH BYTE PARITY ERROR (eroare paritate octet superior).

Aceşti biţi semnalizează cele 3 erori de paritate ce pot apărea la citirea din memoria cache sau la verificarea cîmpului TAG (paritatea cîmpului TAG include şi bitul de validare V); se utilizează, ca şi la memoria centrală, paritatea impară (vezi 3.1.1). În cazul apariţiei unei erori de paritate, cuvîntul respectiv din cache este invalidat (V=0) şi, dacă este autorizată, apare o întrerupere internă cu vectorul 114 (analog cu eroarea de paritate a memoriei centrale).

p.b.15: CPU ABORT (abort procesor); acest bit este setat cînd procesorul a abandonat un ciclu datorită apariţiei unei erori de paritate (validată) la memoria centrală sau la memoria cache.

Registrul este READ ONLY. Poziţiile neutilizate sînt citite 0.

b) *Registrul de control al memoriei cache*
Adresa: 777746



p.b.0: DISABLE TRAPS (invalidare întreruperi); este setat cînd se dorește invalidarea întreruperii în urma unei erori de paritate;
p.b.2: FORCE MISS 0 (invalidare cache, jumătatea inferioară);
p.b.3: FORCE MISS 1 (invalidare cache, jumătatea superioară).
Aceşti doi biţi au rolul de a forţa un ciclu MISS la orice lectură din memoria cache. Bitul 2 forţează ciclu MISS în jumătatea inferioară a memoriei cache (între adresele-index 0 şi 511), iar bitul 3 forţează ciclu MISS în jumătatea superioară (adresele 512 la 1 023). Setînd ambii biţi la 1 se forţează ca toate ciclurile să apeleze memoria centrală, deci practic se invalidează memoria cache.
p.b.6: WRITE WRONG PARITY (scriere paritate greșită); acest bit este utilizat în întreţinere. Dacă este setat, el va forţa scrierea unei parităţi greşite (pară) în cele trei cîmpuri ale cuvîntului cache (TAG, octet superior și octet inferior).
p.b.7: CACHE PARITY ERROR ABORT; acest bit este setat numai în cazul diagnosticării și determină abandonarea unei operaţii cu cache-ul în cazul apariţiei unei erori de paritate.

Acest registru este utilizat pentru validarea sau invalidarea cache-ului, cît și pentru a facilita testarea lui. Poziţiile sale utilizate sînt READ-WRITE; este șters la punerea sub tensiune a calculatorului.

Observaţie: cache-ul minicalculatorului INDEPENDENT 102 F are un singur registru de stare, cu adresa 777746; el poate fi apelat numai prin microprogram cu o comandă specială (MTCH=maintenance cache). Software nu poate fi apelat. Validarea memoriei cache este realizată hardware la iniţializarea sistemului; invalidarea se poate realiza cu ajutorul panoului de comandă, sărind peste microrutina de validare și iniţializare a cache-ului (se lansează microprogramul de emulare de la microadresa 50(8)).

3.4. Procesorul de virgulă mobilă

Procesorul de virgulă mobilă FPP (FLOTANT POINT PROCESOR) este un bloc independent (și opţional) introdus în sistem cu scopul de a mări viteza de execuţie a rutinelor care operează cu numere fracţionare reprezentate în flotant. Aceste rutine sînt întîlnite mai ales în rularea programelor scrise în limbaje de programare evoluate cum ar fi FORTRAN sau BASIC. Procesorul poate executa toate operaţiile aritmetice cu numere în virgulă flotantă, precum și alte operaţii necesare manipulării acestor numere (de exemplu conversiile flotant — întreg și invers).

Modul de reprezentare a numerelor fracţionare în virgulă flotantă a fost descris în paragraful 1.3.2.2.

Caracteristicile FPP sînt:

— viteză mare de execuţie;

— operează cu numere flotante reprezentate în simplă precizie sau în dublă precizie;

— execută anumite operaţii (macrocomenzi) în paralel și asincron faţă de procesorul central;

— moduri de adresare flexibile;

— oferă utilizatorului posibilitatea de a folosi 6 registre — acumulator de cîte 64 de biţi;

— posibilităţi de detecţie și semnalizare erori.

3.4.1. ARHITECTURA FPP

Schema bloc a procesorului de virgulă mobilă (FPP) este prezentată în fig. 3-13.

Principial, FPP funcţionează asemănător cu procesorul central, fiind considerat o parte (extensie) a acestuia. El utilizează aceleași moduri de adresare, cît și facilităţile oferite de sistemul management (bineînţeles dacă există și este activat).

FPP contribuie la execuţia unor instrucţiuni specializate în tratarea numerelor reprezentate în virgulă flotantă; el este astfel proiectat încît să asigure o viteză de lucru sporită în cazul întîlnirii în program a instrucţiunilor de virgulă flotantă. O astfel de instrucţiune se poate executa prin:

— lucrul asociat al procesorului central şi al FPP (fiind utilizate registrele şi operatorul aritmetic FPP specializat în recunoaşterea formatului flotant);

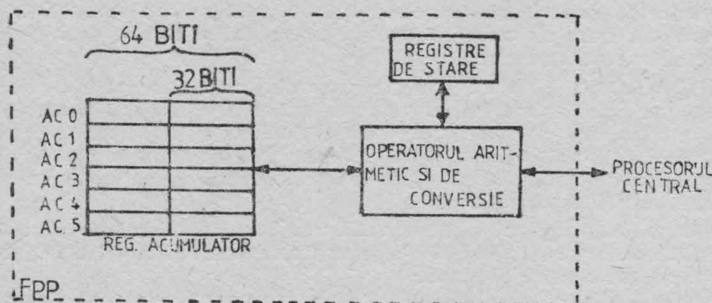


Fig. 3-13 SCHEMA BLOC A FPP

— lucrul independent al FPP, în paralel cu procesorul central (cazul unor macro-comenzi); în această situaţie procesorul central va citi instrucţiunea din memorie, va calcula adresele operanzilor (dacă nu sînt în acumulatori) şi apoi va excita procesorul în virgulă flotantă; FPP va executa independent macro-comanda corespunzătoare, procesorul central aşteptînd sfîrşitul ei exact ca şi cum ar aştepta terminarea unei operaţii de intrare-ieşire.

FPP conţine 6 *registre acumulator* (AC0-AC5) utilizate pentru a stoca local numere (operanzi) în virgulă flotantă. Registrele au lungimea maximă de cîte 64 biţi, fiind deci pregătite să primească numere reprezentate în dublă precizie (sau format dublu D); în cazul simplei precizii (formatul flotant F) vor fi utilizaţi doar 32 biţi din fiecare acumulator. Primele 4 registre — acumulator (AC0-AC3) pot fi utilizate pentru transfer de date între FPP şi registrele generale sau memoria centrală.

Pe lîngă operatorul aritmetic specializat în formatul flotant, FPP dispune şi de trei registre de stare şi de semnalizare a erorilor. Acestea vor fi prezentate în paragraful 3.4.3.

3.4.2. PRECIZIA DE CALCUL A FPP

După cum s-a arătat în paragraful 1.3.2.2., procesorul în virgulă flotantă poate lucra cu numere reprezentate pe două lungimi diferite, în funcţie de precizia dorită:

1) *simplă precizie* sau *modul flotant* F (pentru numere fracţionare reprezentate pe 32 biţi), respectiv *modul întreg I* (pentru numere întregi reprezentate pe 16 biţi);

2) *dublă precizie* sau *modul dublu* D (pentru numere fracţionare reprezentate pe 64 de biţi), respectiv *modul lung L* (pentru numere întregi reprezentate pe 32 de biţi).

FPP posedă cîte un operator aritmetic pentru fiecare cîmp al formatului flotant. Există deci trei operatori: unul pentru mantisă, unul pentru exponent şi al treilea (cel mai simplu) pentru semnul mantisei. Operatorul pe exponent (caracteristică) va detecta cazurile de depăşire flotantă superioară (overflow), sau inferioară (underflow). Operatorul pe mantisă (partea fracţionară a numărului), decide precizia de calcul în funcţie de formatul folosit şi de rotunjirea rezultatului.

O instrucţiune sau o operaţie este numită *exactă* dacă rezultatul este identic cu cel care s-ar obţine în urma unui calcul cu precizie infinită, folosind, bineînţeles, aceiaşi operanzi. Numerele care au caracteristica 0 sînt interpretate în operaţiile aritmetice ca 0 exact (vezi 1.3.2.2.). Pentru toate operaţiile aritmetice (exceptînd îm-

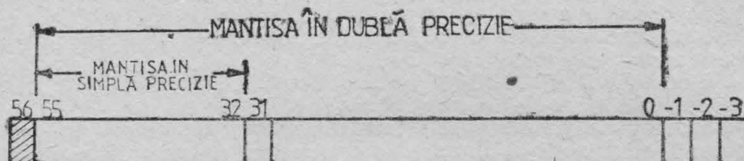
părțirea), un operand 0 implică un rezultat exact, deci instrucțiunea este considerată exactă; același lucru este valabil și în cazul împărțirii dacă operandul 0 este deîmpărțit; dacă 0 este împărțitor, operația este nedefinită și va fi semnalată o eroare.

În cazul numerelor flotante nenule, partea fracționară este binar-normalizată (vezi 1.3.2.2.). În funcție de dorința utilizatorului, partea fracționară a rezultatului poate fi obținută *rotunjită* sau *nerotunjită (trunchiată)*.

Partea fracționară a numărului flotant conține 24 biți în modul F (simplă precizie) și 56 biți în modul D (dublă precizie). Registrele interne de lucru, cît și operatorul pe mantisă, au 60 de biți pentru prelucrarea părții fracționare. Bitul cel mai semnificativ este rezervat depășirii aritmetice. Rezultă că, intern, rămîn 35 biți de rezervă pentru modul F și 3 biți de rezervă pentru modul D; dintre aceștia, doar doi biți sînt necesari și suficienți pentru a realiza operația de rotunjire.

Rezultatul trunchiat se obține prin înlăturarea biților de rezervă, fără o prelucrare suplimentară a celorlalți biți utili aparținînd mantisei. Rezultatul trunchiat (nerotunjit) va avea deci o eroare limitată la 1 din cifra cea mai puțin semnificativă (LSB) a mantisei.

În cazul rezultatului rotunjit, eroarea este limitată la 1/2 din LSB, prin prelucrarea bitului cel mai puțin semnificativ al mantisei în raport cu valoarea biților de rezervă; corecția se realizează plecînd de la bitul cel mai semnificativ pierdut prin trunchiere, numit bit de rotunjire. Intern, mantisa este prelucrată astfel:



p.b.56: bitul ascuns (hidden bit)
p.b. —1, —2, —3: biți de rezervă

1° în cazul simplei precizii, mantisa ocupă pozițiile 32—55; ceilalți biți (—3, —2, ..., +31) sînt de rezervă, fiind înlăturați pentru obținerea rezultatului trunchiat; bitul 31 este bitul de rotunjire;

2° în cazul dublei precizii, mantisa ocupă pozițiile 0—55; doar biții de rezervă —3, —2, și —1 sînt înlăturați la trunchiere, bitul de rotunjire fiind p.b. —1.

În urma unei operații aritmetice un rezultat este numit exact dacă biții de rezervă (înlăturați prin trunchiere) sînt nuli. Valoarea rotunjită a rezultatului se obține din valoarea trunchiată în modul următor:

— dacă bitul de rotunjire este 0, valoarea rotunjită este identică cu cea trunchiată;

— dacă bitul de rotunjire este 1, valoarea rotunjită se obține incrementînd cu 1 valoarea trunchiată.

Rezultă următoarele:

a) dacă rezultatul este exact, atunci valoarea rotunjită=valoarea trunchiată= valoare exactă;

b) dacă rezultatul nu este exact, atunci mărimea sa:
— descrește întotdeauna prin trunchiere;
— descrește prin rotunjire dacă bitul de rotunjire este 0;
— crește prin rotunjire dacă bitul de rotunjire este 1.

Apariția unei depășiri superioare (overflow) sau inferioare (underflow) este condiție de eroare. Rezultatul operației nu poate fi exact memorat, deoarece exponentul depășește capacitatea celor 8 biți care-i sînt rezervați. FPP poate genera o întrerupere pentru fiecare caz de eroare, spre procesorul central; întreruperea poate fi validată sau invalidată de utilizator (cu ajutorul registrelor de stare FPP). Studiind tipul de eroare care a generat întreruperea, precum și rezultatul (incorect) al operației, utilizatorul poate reconstitui (de cele mai multe ori) cauza erorii.

În cazul unei depășiri inferioare, înlocuirea răspunsului corect cu zero, constituie o rezolvare acceptabilă în majoritatea aplicațiilor; acest lucru se întîmplă

dacă întreruperea corespunzătoare este invalidată, deci FPP n-are posibilitatea de a semnaliza eroarea. În cazul unei depășiri superioare, însă, rezolvarea problemei nu mai este atît de simplă. Oricum, trebuie reținut că partea fracționară este neafectată de apariția unei depășiri; de asemenea, din exponentul (caracteristica) rezultat depășit se memorează cei 8 biți mai puțin semnificativi.

3.4.3. REGISTRELE DE STARE

3.4.3.1. Registrul de stare și control FPS



p.b.0: FC = FLOATING CARRY (transport flotant); indicatorul este setat dacă în ultima operație executată a apărut un transport către poziția cea mai semnificativă; acest lucru se poate întîmpla numai în cazul operațiilor de conversie flotant — întreg (la depășirea formatului întreg).

p.b.1: FV = FLOATING OVERFLOW (depășire flotantă); indicatorul este setat dacă la ultima operație a rezultat un exponent (caracteristică) depășit.

p.b.2: FZ = FLOATING ZERO (zero flotant); indicatorul este setat dacă rezultatul ultimei operații executate a fost zero.

p.b.3: FN = FLOATING NEGATIVE (negativ flotant); indicatorul este setat dacă la ultima operație a rezultat o mantisă negativă.

p.b.5: FT = FLOATING CHOP MODE (mod trunchiat); cînd acest bit este 1, rezultatul oricărei operații aritmetice este trunchiat (nerotunjit); cînd bitul este zero se execută rotunjirea rezultatului (vezi 3.4.2.).

p.b.6: FL = FLOATING LONG INTEGER MODE (mod întreg lung); acest bit este activ în conversiile flotant — întreg și invers. Dacă FL=1, întregul selectat este considerat în dublă precizie (pe 32 biți); dacă FL=0, formatul numărului întreg este restrîns la 16 biți (simplă precizie).

p.b.7: FD = FLOATING DOUBLE PRECISION MODE (mod dublă precizie); acest bit determină precizia utilizată de FPP în lucrul cu numere flotante. Dacă FD=1, este selectată dubla precizie (numere flotante pe 4×16=64 biți); dacă FD=0, este selectată precizia simplă (numere flotante pe 2×16=32 biți).

p.b.8: FIC = INTERRUPT ON INTEGER CONVERSION ERROR (întrerupere pe eroare de conversie în întreg); acest bit validează întreruperea către procesorul central la apariția unei erori de conversie din flotant în întreg (este depășit formatul întreg lung sau scurt precizat prin indicatorul FL). Dacă FIC=1 și conversia flotant — întreg depășește, atunci la destinație se depune 0 și apare o întrerupere; dacă FIC=0 rezultatul operației depășite este același, dar nu mai apare întrerupere.

p.b.9: FIV = INTERRUPT ON OVERFLOW (întrerupere la depășire superioară); acest bit validează întreruperea care apare în urma unei depășiri superioare a exponentului (exponent mai mare decît +127 (în zecimal)). Dacă FIV=1, în caz de depășire superioară apare o întrerupere; partea fracționară (mantisa) a rezultantului va fi corectă; caracteristica reprezentată va fi mai mică decît cea reală; pentru a afla caracteristica reală trebuie adunat un multiplu de 400 (octal). Dacă FIV=0, nu mai apare întrerupere în cazul unei depășiri superioare; la destinație se memorează rezultatul 0 exact (cu unele excepții discutate la descrierea instrucțiunilor).

p.b.10: FIU = INTERRUPT ON UNDERFLOW (întrerupere la depășire inferioară); acest bit validează întreruperea care apare în urma unei depășiri inferioare a exponentului (exponent rezultat mai mic decît —127 (în zecimal)). Dacă FIU=1, în caz de depășire inferioară apare o întrerupere; partea fracționară a rezultatului este corectă; caracteristica reprezentată este mai mare decît cea reală; pentru a afla caracteristica reală trebuie scăzut un multiplu de 400 (octal); excepție în cazul caracteristicii 0 care este corectă. Dacă FIU=0 și operația se

încheie cu depășire inferioară, nu apare întrerupere; la destinație se memorează rezultatul 0 exact.

p.b.11: FIUV = INTERRUPT UNDEFINED VARIABLE (întrerupere pe variabilă nedefinită); acest bit validează întreruperea ce apare la întîlnirea unei variabile nedefinite (—0, vezi 1.3.2.2.). Dacă FIUV=1, apare o întrerupere la încărcarea din memorie a unui operand —0; în general, întreruperea este declanșată înaintea execuției, cu excepția instrucțiunilor NEG și ABS, cazuri în care întreruperea apare după execuție. Întreruperea nu este declanșată dacă operandul —0 este luat dintr-un registru acumulator. Dacă FIUV=0, variabila nedefinită —0 poate fi încărcată și utilizată în FPP, fără apariția vreunei întreruperi. Dacă —0 apare ca rezultat al unei operații, FPP nu-l va memora decît dacă întreruperea respectivă este validată (FIUV=1).

p.b.14: FID = INTERRUPT DISABLE (invalidare întreruperi); dacă FID=1, orice întrerupere a FPP este invalidată. Este de reținut faptul că, dacă o întrerupere individuală este validată, atunci numai întreruperea este inhibată (restul operațiilor rămîn identice). Acest bit este în primul rînd o facilitare pentru întreținere (testarea FPP); normal el trebuie să fie șters (mai ales dacă se dorește să se asigure că memorarea lui —0 este întotdeauna acompaniată de o întrerupere). Bineînțeles că acțiunea tuturor biților de validare a întreruperilor individuale (FIC, FIV, FIU și FIUV) depinde și de starea acestui bit.

p.b.15: FER = FLOATING ERROR (eroare în flotant); bitul este setat ori de cîte ori este detectată o condiție de eroare în FPP. Erorile posibile sînt:
— încercarea de împărțire cu zero;
— cod ilegal de instrucțiune în virgulă flotantă;
— orice eroare căreia îi corespunde o întrerupere (eroare de conversie din flotant în întreg, depășire flotantă superioară, depășire flotantă inferioară, variabilă nedefinită).

Setarea indicatorului de eroare este independentă de starea bitului FID. Este de notat faptul că FPP nu resetează niciodată bitul FER; acest indicator poate fi șters numai cu instrucțiunea RESET sau cu instrucțiunea în virgulă mobilă LDFPS (încărcare registru de stare FPP).

3.4.3.2. **Registrele de eroare**

În mod normal, orice eroare detectată de FPP este acompaniată de o întrerupere spre procesorul central. După cum s-a văzut din paragraful anterior, întreruperile pot fi invalidate poziționînd corespunzător anumiți biți din registrul de stare și control FPS.

VECTORUL DE ÎNTRERUPERE=244

Fiecărei erori îi corespunde un cod pentru recunoastere; acest cod este memorat automat în registrul FEC (FLOATING EXCEPTION CODE) la apariția erorii respective. Intern, registrul FEC are 4 biți cadrați la dreapta în raport cu un cuvînt, deci ocupînd pozițiile mai puțin semnificative. Cele 6 posibile erori și codurile corespunzătoare (octal) sînt următoarele:

1) cod ilegal de instrucțiune în virgulă flotantă:	2
2) încercare de împărțire cu zero:	4
3) eroare de conversie flotant — întreg:	6
4) depășire flotantă superioară:	10
5) depășire flotantă inferioară:	11
6) variabilă nedefinită:	12

Adresa logică a instrucțiunii care a provocat eroarea este memorată în registrul FEA (FLOATING EXCEPTION ADDRESS); acesta are lungimea de un cuvînt.

Registrele FEC și FEA sînt actualizate ori de cîte ori apare o condiție de eroare și întreruperea corespunzătoare este validată (în cazul celor 4 erori cu întreruperi mascabile). Invalidarea unei întreruperi individuale va inhiba actualizarea registrelor FEC și FEA la apariția erorii corespunzătoare; starea bitului FID, însă, nu influențează actualizarea registrelor de eroare. De asemenea, FEC și FEA nu sînt actualizate dacă nu apare condiție de eroare, ele „înghețînd" cu datele ultimei erori apărute.

Utilizatorul poate examina registrele de eroare numai după ce le-a transferat conţinutul în memoria centrală a calculatorului cu ajutorul instrucţiunii STST (STORE STATUS). Nu există nici o instrucţiune cu ajutorul căreia să se poată scrie în FEC şi FEA, deci, din punctul de vedere al utilizatorului, aceste registre sînt READ ONLY (citeşte numai).

3.4.4. CLASIFICAREA ŞI FORMATUL INSTRUCŢIUNILOR DE VIRGULĂ MOBILĂ

3.4.4.1. **Clasificarea instrucţiunilor de virgulă mobilă.** Din punctul de vedere al operaţiilor pe care le realizează, instrucţiunile de virgulă mobilă pot fi clasificate în trei mari categorii:

1) *instrucţiuni aritmetice şi logice;* cu ajutorul acestor instrucţiuni se pot executa o serie de operaţii logice asupra numerelor reprezentate în flotant, precum şi cele 4 operaţii aritmetice fundamentale (adunare, scădere, înmulţire şi împărţire).

2) *instrucţiuni de încărcare, memorare şi conversie;* aceste instrucţiuni realizează transferul de date între registrele acumulator ale FPP şi memoria centrală:

acumulator ← memorie ⇒ încărcare;

acumulator → memorie ⇒ memorare (stocare).

Unele instrucţiuni permit ca odată cu transferul să se realizeze şi conversii din simplă în dublă precizie (sau invers), precum şi conversii din flotant în întreg.

3) *instrucţiuni de stare;* aceste instrucţiuni asigură interfaţa dintre utilizator şi registrele de stare a FPP.

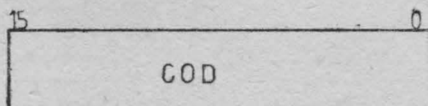
3.4.4.2. **Formatul instrucţiunilor de virgulă flontantă.** O instrucţiune în virgulă flotantă ocupă un cuvînt (16 biţi) în memoria centrală a calculatorului; ea poate fi urmată de un operand imediat sau un index, la fel ca şi în cazul instrucţiunilor de bază prezentate în capitolul 2. O instrucţiune în virgulă flotantă se deosebeşte de celelalte prin faptul că primele 4 cifre binare cele mai semnificative ale codului său sînt 1. Deci, în cifre octale, codul unei instrucţiuni în virgulă mobilă va avea forma:

17XX XX,

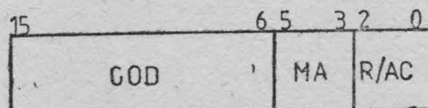
unde X este o cifră octală oarecare.

Formatul instrucţiunilor depinde de numărul operanzilor:

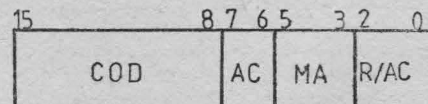
a) *instrucţiuni fără operanzi (de control):*



b) *instrucţiuni cu un operand:*



c) *instrucţiuni cu doi operanzi:*



unde: COD = codul instrucţiunii;
MA = modul de adresare a operandului care se poate afla oriunde în memoria centrală sau în acumulator;

R/AC = numărul registrului general utilizat pentru aflarea operandului sau numărul acumulatorului în care se află operandul (AC0—AC5); este interpretat în conjuncție cu MA;

AC = numărul acumulatorului în care se află unul dintre operanzi (AC0—AC3).

Modurile de adresare sînt descrise în paragraful următor (3.4.4.3.).

Observații:

— în cazul instrucțiunilor cu operanzi, un singur operand poate fi oriunde în memoria centrală a calculatorului; pentru localizarea sa se utilizează un mod de adresare MA și un registru general R din procesorul central; după cum se va vedea mai departe, dacă modul de adresare este 0, atunci operandul se află într-un registru acumulator AC;

— în cazul instrucțiunilor cu doi operanzi, unul dintre aceștia se află întotdeauna într-unul dintre primele patru registre acumultor din FPP (AC0—AC3); se observă că AC are doar 2 biți. Cu ajutorul acestor instrucțiuni cu doi operanzi se realizează legătura dintre acumulatoarele FPP și memoria centrală a calculatorului; din această cauză AC0—AC3 se mai numesc și acumulatori de legătură sau de intrare-ieșire;

— atenție la scrierea cifrei octale definită de biții 6—8 din codul instrucțiunii! Bitul cel mai semnificativ (p.b.8) aparține codului, ceilalți 2 biți marcînd numărul acumulatorului de legătură folosit; în consecință, s-a adoptat următoarea notație pentru scrierea acestei cifre octale în codul instrucțiunii:

AC dacă p.b.8=0;
(AC+4) dacă p.b.8=1.

3.4.4.3. **Modurile de adresare.** Procesorul de virgulă mobilă folosește aceleași moduri de adresare ca și procesorul central. Adresa unui operand sursă sau destinație se află utilizînd unul dintre cele 8 moduri de adresare (MA), precum și unul dintre cele 8 registre generale R ale procesorului central. Modurile de adresare sînt aceleași ca la procesorul central (prezentate în paragraful 2.1.4.), cu excepția modului 0, care depinde de tipul instrucțiunii:

a) *instrucțiuni care folosesc moduri de adresare pentru FPP;* în acest caz, modul de adresare 0 vizează un registru acumulator din FPP;

b) *instrucțiuni care folosesc moduri de adresare pentru procesorul central* (caz în care modul de adresare 0 este cel obișnuit, vizînd deci un registru general din procesorul central).

Cele opt moduri de adresare sînt:

0 : direct (acumulator sau registru general)
1 : indirect
2 : autoincrement
3 : autoincrement-indirect
4 : autodecrement
5 : autodecrement-indirect
6 : indexat
7 : indexat-indirect.

În cazul modului de adresare 0 flotant (direct acumulator), operandul se află în acumulatorul specificat de biții 0—2 ai instrucțiunii. Pot fi utilizate toate cele 6 acumulatoare din FPP (AC0—AC5); dacă se adresează acumulatoarele inexistente AC6 sau AC7, atunci FPP semnalizează eroarea cu codul 2 în registrul FEC (cod ilegal de instrucțiune în virgulă flotantă).

Toate celelalte moduri de adresare (1—7) utilizează un registru general din procesorul central pentru găsirea operanzilor; în aceste cazuri operanzii se află în memoria centrală a calculatorului, deci vor avea loc operații de intrare-ieșire pe bus. Lungimea operanzilor citiți din memorie sau stocați în memorie este impusă de starea indicatorilor FD și FL din registrul FPS (vezi 3.4.3.1.) și anume:

— în cazul unui operand flotant, el va fi considerat pe două cuvinte dacă FD=0 (simplă precizie) sau pe patru cuvinte dacă FD=1 (dublă precizie);

— în cazul unui operand întreg, el va fi considerat pe un cuvînt dacă FL=0 (simplă precizie) sau pe două cuvinte dacă FL=1 (dublă precizie).

Prin adresa operandului se înţelege adresa cea mai mică de memorie corespunzătoare celor n cuvinte pe care se desfăşoară operandul respectiv (n=1, 2, 4); este vorba deci de adresa cuvîntului care conţine caracteristica (exponentul) numărului sau de adresa cuvîntului care conţine partea mai semnificativă a numărului.

În cazul modurilor de adresare 2 sau 4, autoincrementarea (respectiv autodecrementarea) conţinutului registrului R utilizat (care conţine adresa operandului) se face în funcţie de lungimea numărului flotant sau întreg exploatat, deci:

— cu 2 în cazul FL=0 (modul întreg scurt);

— cu 4 în cazul FL=1 (modul întreg lung) sau FD=0 (modul flotant simplu);

— cu 10 (octal) în cazul FD=1 (modul flotant dublu).

Excepţie face cazul modului de adresare imediat 27 (modul de adresare 2 cu PC); în această situaţie doar un cuvînt (16 biţi) este încărcat sau stocat, localizat imediat după instrucţiune; autoincrementarea se face cu 2 (indiferent de lungimea operandului).

În cazul modului de adresare 4, trebuie ţinut cont că decrementarea se execută înainte de adresare operandului, deci, registrul R va trebui să conţină adresa care urmează după ultimul cuvînt al numărului vizat.

Pentru modurile indirecte 3 sau 5, incrementările (respectiv decrementările) se fac cu 2, deoarece registrele utilizate R nu mai conţin adrese de operanzi ci adrese de adrese (care ocupă un singur cuvînt în memorie).

3.4.4.4. **Notaţii şi simboluri utilizate.** Modul de interpretare a operanzilor depinde de precizia sub care se lucrează la un moment dat, deci de starea indicatorilor FD şi FL din registrul de stare al FPP (vezi 3.4.3.1.). Execuţia unei anumite instrucţiuni este influenţată de precizia de calcul (care defineşte, de fapt, lungimea operanzilor); din această cauză, unui anumit cod de instrucţiune îi poate corespunde mai multe mnemonici (nume); scopul este de a discrimina preciziile (sau modurile de reprezentare a operanzilor). Numele instrucţiunilor care au acelaşi cod, dar pot opera în diferite precizii diferă între ele prin ultima literă sau prin ultimele două litere. S-a adaptat notaţia:

NUME(X/Y),

care indică faptul că instrucţiunea are mnemonica NUMEX sau NUMEY, în funcţie de precizia definită a operanzilor.

Alte notaţii:

SS/DD=operand sursă/destinaţie adresat utilizînd modurile de adresare ale procesorului central (vezi 2.1.4.); el se poate afla, deci, în memoria centrală sau în registrele generale ale CPU;

FS/FD=operand sursă/destinaţie adresat utilizînd modurile de adresare ale procesorului de virgulă mobilă (vezi 3.4.4.3.); el se poate afla, deci, în memoria centrală sau în registrele acumulatori din FPP;

AC=registrul acumulator utilizat de instrucţiune;

A/(A+4)=cifra octală definită de biţii 6—8 din codul instrucţiunii (bitul 8 aparţine codului operaţiei şi poate fi 0 în cazul A sau 1 în cazul (A+4); A indică numărul acumulatorului utilizat);

(X)	=	conţinutul operandului X;
ABS(X)	=	valoarea absolută a cantităţii X;
EXP(X)	=	caracteristica numărului flotant X;
X/Y	=	X sau (respectiv) Y;
$<$/$\leqslant$	=	mai mic/mai mic sau egal;
$>$/$\geqslant$	=	mai mare/mai mare sau egal.

3.4.5. INSTRUCŢIUNI ARITMETICE ŞI LOGICE

3.4.5.1. Instrucţiunea CLR(F/D)

— *mnemonică şi cod:*	CLRF CLRD	1704FD

— *nume:* clear floating/double (șterge destinație mod flotant/dublu)
— *operație:* **FD←0 exact.**

Destinația este încărcată cu 0 pe lungimea de 2 cuvinte (CLRF dacă FD=0), respectiv 4 cuvinte (CLRD dacă FD=1). Instrucțiunea este exactă. Nu poate apare nici o eroare și nici o întrerupere.
— *poziționarea indicatorilor de condiții:*

FN=0.
FZ=1.
FV=0.
FC=0.

3.4.5.2. **Instrucțiunea ABS(F/D)**

— *mnemonică și cod:*

ABSF ABSD	1706FD

— *nume:* make absolute floating/double (încarcă valoare absolută mod flotant/dublu).
— *operație:* **FD←(FD) dacă (FD)>0, sau**
FD← —(FD) dacă (FD)<0, sau
FD← 0 exact dacă EXP(FD)=0.

Conținutul destinației este setat la valoarea sa absolută (practic se forțează semnul mantisei la 0). Instrucțiunea este exactă. Nu poate apărea depășire inferioară sau superioară. Dacă destinația citită din memorie este — 0 (variabila nedefinită) și FIUV=1 (întrerupere corespunzătoare validată), atunci este memorat 0 exact; după execuția instrucțiunii este declanșată întreruperea.
— *poziționarea indicatorilor de condiții:*

FN=0.
FZ=1 dacă EXP(FD)=0; în rest, FZ=0.
FV=0.
FC=0.

3.4.5.3. **Instrucțiunea NEG(F/D)**

— *mnemonică și cod:*

NEGF NEGD	1707FD

— *nume:* negate floating/double (negare destinație mod flotant/dublu)
— *operație:* **FD← — (FD) dacă EXP (FD)≠0, altfel:**
FD←0 exact.

Se neagă conținutul destinației (practic se complementează valoarea bitului de semn a mantisei). Instrucțiunea este exactă. Nu poate apărea depășire superioară sau inferioară. Cazul apariției variabile nedefinite —0 este tratat identic ca la instrucțiunea precedentă: ABS (F/D).
— *poziționarea indicatorilor de condiții:*

FN=1 dacă (FD) rezultat este negativ; altfel, FN=0.
FZ=1 dacă EXP(FD)=0; altfel FZ=0.
FV=0.
FC=0.

3.4.5.4. **Instrucţiunea TST(F/D)**

— *mnemonică şi cod:*	TSTF TSTD	1705FD

— *nume:* test floating/double (test destinaţie mod flotant/dublu).
— *operaţie:* FD←(FD).
Indicatorii de condiţii din FPP sînt setaţi conform conţinutului destinaţiei. Instrucţiunea este exactă. Conţinutul destinaţiei nu se modifică. Dacă destinaţia este —0 şi FIUV=1 apare întrerupere după execuţie.

— *poziţionarea indicatorilor de condiţii:*

FN=1 dacă (FD)<0, altfel FN=0.
FZ=1 dacă EXP(FD)=0, altfel FZ=0.
FV=0.
FC=0.

3.4.5.5. **Instrucţiunea CMP(F/D)**

— *mnemonică şi cod:*	CMPF CMPD	173(A+4)FS

— *nume:* comparare floating/double (comparare în mod flotant/dublu).
— *operaţie:* (FS)—(AC).
Se compară conţinutul operandului sursă cu cel al acumulatorului specificat şi sînt poziţionali corespunzător indicatorii de condiţii din FPP. Dacă un operand are caracteristica 0, atunci el este tratat ca 0 exact. Cei doi operanzi rămîn nemodificaţi. Instrucţiunea este exactă. Dacă apare —0 şi FIUV=1 se generează o întrerupere înaintea execuţiei instrucţiunii.
— *poziţionarea indicatorilor de condiţii:*

FN=1 dacă (FS) — (AC)<0, altfel FN=0.
FZ=1 dacă (FS) — (AC)=0 (sau cei doi operanzi sînt egali), altfel FZ=0.
FV=0.
FC=0.

3.4.5.6. **Instrucţiunea ADD(F/D)**

— *mnemonică şi cod:*	ADDF ADDD	172AFS

— *nume:* add floating/double (adunare în mod flotant/dublu).
— *operaţie:* AC←(AC)+(FS).
Se adună conţinutul operandului sursă cu cel al acumulatorului specificat AC, rezultatul depunîndu-se în acumulator. Rezultatul este trunchiat sau rotunjit conform indicatorului FT din registrul de stare FPS. Cazul apariţiei unei depăşiri flotante superioare (overflow) sau inferioare (underflow) este tratat în modul următor:
1) dacă întreruperile corespunzătoare sînt invalidate (FIV=0 sau FIU=0), atunci în AC este forţat rezultatul zero exact;

2) dacă întreruperile corespunzătoare sînt validate (FIV=1 sau FIU=1), atunci FPP va genera o întrerupere și în AC se va depune rezultatul eronat; mantisa este corectă; caracteristica este mai mică de 400 (octal) în caz de overflow; ea este mai mare de 400 (octal) în caz de underflow, cu excepția caracteristicii 0, care este corectă (vezi și FIV, FIU din paragraful 3.4.3.1.).

Apariția unui operand —0 este însoțită de o întrerupere înaintea execuției instrucțiunii dacă FIUV=1. Ca rezultat, variabila nedefinită —0 poate apărea numai în conjuncție cu o depășire (underflow sau overflow); ea va fi memorată în AC numai dacă întreruperea corespunzătoare este validată (FIUV=1).

— *poziționarea indicatorilor de condiții:*
FN=1 dacă rezultatul este negativ, altfel FN=0.
FZ=1 dacă rezultatul este nul, altfel FZ=0.
FV=1 dacă apare depășire flotantă superioară (overflow), altfel FV=0.
FC=0.

3.4.5.7. **Instrucțiunea SUB(F/D)**

— *mnemonică și cod:*

SUBF SUBD	173AFS

— *nume:* substract floating/double (scădere în mod flotant/dublu).
— *operație:* **AC←(AC)—(FS).**

Se scade conținutul sursei din acumulatorul specificat AC, rezultatul depunîndu-se în acumulator. Rezultatul este trunchiat sau rotunjit conform indicatorului FT din registrul de stare FPS. Cazurile de eroare (depășire flotantă superioară sau inferioară, apariția variabilei nedefinite —0) sînt tratate ca și la instrucțiunea de adunare (vezi 3.4.5.6.).

— *poziționarea indicatorilor de condiții:*

FN=1 dacă rezultatul este negativ, altfel FN=0.
FZ=1 dacă rezultatul este nul, altfel FZ=0.
FV=1 dacă apare depășire flotantă superioară (overflow), altfel FV=0.
FC=0.

3.4.5.8. **Instrucțiunea MUL(F/D)**

— *mnemonică și cod:*

MULF MULD	171AFS

— *nume:* multiply floating/double (înmulțire în mod flotant/dublu).
— *operație:* **AC←(AC)×(FS).**

Se înmulțește conținutul acumulatorului cu operandul sursă, rezultatul depunîndu-se în acumulatorul specificat. Dacă cel puțin un operand are caracteristica 0, atunci în AC se depune 0 exact. Rezultatul este trunchiat sau rotunjit conform indicatorului FT din registrul de stare FPS. Cazurile de eroare (depășire flotantă sau variabilă nedefinită) sînt tratate ca și la instrucțiunea de adunare (vezi 3.4.5.6.).

— *poziționarea indicatorilor de condiții:*

FN=1 dacă rezultatul este negativ, altfel: FN=0.
FZ=1 dacă rezultatul este nul, altfel: FZ=0.
FV=1 dacă apare depășire superioară (overflow), altfel FV=0.
FC=0.

3.4.5.9. **Instrucţiunea DIV(F/D)**

— *mnemonică şi cod:*	DIVF DIVD	174(A+4)FS

— *nume:* divide floating double (împărţire în mod flotant/dublu).

— *operaţie:* **AC←(AC)/(FS).**

Se împarte conţinutul acumulatorului specificat la operandul sursă; rezultatul se depune în acumulator. Dacă unul dintre operanzi are caracteristica 0, el este interpretat ca 0 exact. Dacă EXP (AC)=0, atunci rezultatul operaţiei este 0 exact (AC←0). Dacă EXP (FS)=0, atunci operaţiunea este abandonată (încercare de împărţire cu 0); conţinutul acumulatorului nu se modifică; FPP va genera o întrerupere şi va seta în registrul FEC eroarea cu codul 4. Rezultatul este trunchiat sau rotunjit conform indicatorului FT din registrul de stare FPS. Cazurile celorlalte erori (depăşire flotantă sau variabilă nedefinită) sînt tratate ca şi la instrucţiunea de adunare (vezi 3.4.5.6.).

— *poziţionarea indicatorilor de condiţii:*

FN=1 dacă rezultatul este negativ, altfel FN=0.
FZ=1 dacă EXP (AC)=0, altfel FZ=0.
FV=1 dacă apare depăşire flotantă superioară (overflow), altfel FV=0.
FC=0.

3.4.5.10. **Instrucţiunea MOD(F/D)**

— *mnemonică şi cod:*	MODF MODD	171(A+4)FS

— *nume:* multiply and integerize floating/double (înmulţire în mod flotant/dublu cu selecţie de parte întreagă şi parte fracţionară).

— *operaţie:* instrucţiunea calculează produsul dintre conţinutul acumulatorului specificat AC şi operandul sursă, separînd rezultatul în două părţi: partea întreagă (N) şi partea fracţionară (f); aceste două părţi vor fi memorate ca două numere distincte reprezentate în virgulă flotantă. Execuţia instrucţiunii va implica următoarele operaţii:

a) PROD=(AC)×(FS).

b) Produsul PROD are valoarea absolută binară:

$$\mathrm{ABS(PROD)}=2^{k}\cdot g,$$

unde K=EXP(PROD)—200 (octal) şi $1/2 \leqslant g < 1$ (vezi reprezentarea numerelor în flotant, paragraful 1.3.2.2.). Totodată se poate scrie că PROD=N+f, unde:

N=INT(PROD)=partea întreagă a numărului PROD;

f=PROD—N=partea fracţionară a numărului PROD, cu $0 \leqslant f < 1$. N şi f au acelaşi semn cu PROD.

c) Rezultatul instrucţiunii se obţine astfel:

c1) dacă AC este un acumulator cu număr par (AC0 sau AC2), atunci f este depus în AC, iar N este depus în AC+1 (AC1 sau, respectiv, AC3);

c2) dacă AC este un acumulator cu număr impar atunci f este depus în AC, iar N este pierdut. Se poate deci scrie că:

$$AC_{v}1 \leftarrow N \text{ şi apoi}$$

$$AC \leftarrow f,$$

unde notaţia v indică operaţia logică SAU.

În continuare se consideră n=24 pentru simplă precizie și n=56 pentru dublă precizie (se observă că n reprezintă numărul de biți ai mantisei, plus un bit de depășire).

Avînd în vedere că N și f sînt la rîndul lor reprezentate în flotant, se disting următoarele 5 cazuri:

1) PROD depășește superior (overflow); deosebim două subcazuri:

1a) FIV=1 (întreruperea corespunzătoare este validată):

$AC_v1 \leftarrow N$ (trunchiat la n biți)

$AC \leftarrow 0$ exact.

EXP(N) este mai mic de 400 (octal) și un eventual —0 poate fi depus în AC_v1.

1b) FIV=0 (întreruperea corespunzătoare este invalidată):

$AC_v1 \leftarrow 0$ exact

$AC \leftarrow 0$ exact,

iar —0 nu va fi niciodată stocat.

2) ABS(PROD)$\geqslant 2^n$, dar nu apare depășire superioară:

$AC_v1 \leftarrow N$ (trunchiat la n biți)

$AC \leftarrow 0$ exact.

Semnul și caracteristica lui N sînt corecte, dar bitul cel mai puțin semnificativ al lui N (bitul de paritate) este pierdut.

3) $1 \leqslant$ ABS(PROD) $< 2^n$:

$AC_v1 \leftarrow N$

$AC \leftarrow f$.

Partea întreagă N este exactă. Partea fracționară f este normalizată și apoi trunchiată sau rotunjită conform indicatorului FT din registrul de stare FPS.

4) ABS(PROD)<1, fără depășire inferioară:

$AC \leftarrow f$.

$AC_v1 \leftarrow 0$ exact

Normal, partea întreagă este exact 0. Partea fracționară f este prelucrată ca la cazul 3).

5) PROD depășește inferior (underflow); avem două subcazuri:

5a) FIV=1 (întreruperea corespunzătoare este validată):

$AC_v1 \leftarrow 0$ exact

$AC \leftarrow f$.

EXP(AC) va fi mai mare de 400 (octal) cu excepția EXP(AC)=0, caz în care e corect. Va apare o întrerupere, iar —0 eventual rezultat va fi depus în AC.

5b) FIV=0 (întreruperea corespunzătoare este invalidată):

$AC_v1 \leftarrow 0$ exact

$AC \leftarrow 0$ exact.

Variabila nedefinită —0 nu va fi memorată.

Dacă unul dintre operanzi este —0 și FIUV=1 va apărea o întrerupere înaintea execuției instrucțiunii.

— *poziţionarea indicatorilor de condiţii* se face conform rezultatului obţinut în AC:

FN=1 dacă (AC)<0, altfel FN=0.
FZ=1 dacă (AC)=0, altfel FZ=0.
FV=1 dacă PROD depăşeşte superior (overflow), altfel FV=0.
FC=0.

Ca o aplicaţie a acestei instrucţiuni este prezentată conversia din binar în zecimal a unei fracţii subunitare: algoritmul următor (fig. 3-14), folosind instrucţiunea MOD, va genera cifrele zecimale Z(1), Z(2), ... de la stînga la dreapta (după punctul zecimal); fie X numărul de convertit cu 0⩽ABS(X)<1. Considerăm că, iniţial: Z(1)=Z(2)= ... =0.

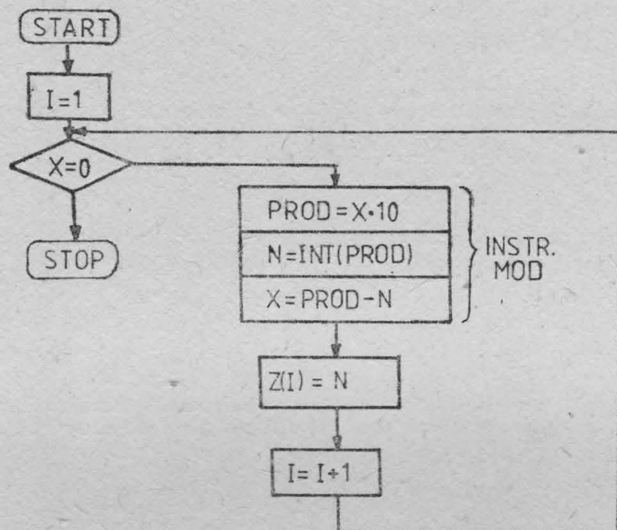


Fig.3-14 EXEMPLU DE UTILIZARE A INSTRUCŢIUNII MOD

Se observă că după fiecare ciclu se obţine o cifră zecimală Z(I); înmulţirea cu 10 a fracţiei X corespunde cu deplasarea ei spre stînga (faţă de punctul zecimal) cu un anumit număr de poziţii. Partea întreagă a produsului va fi egală tocmai cu reprezentarea binară a cifrei zecimale Z(I); partea fracţionară devine X-ul pentru următorul ciclu. Operaţia se încheie după ce s-au epuizat toate cifrele semnificative ale lui X (X=0); ea se poate încheia şi după un număr dat de cicluri.

3.4.6. INSTRUCŢIUNI DE ÎNCĂRCARE, MEMORARE ŞI CONVERSIE

3.4.6.1. Instrucţiunea LD(F/D)

— *mnemonică şi cod:*	LDF LDD	172(A+4)FS

— *nume:* load floating/double (încărcare mod flotant/dublu).

— *operație:* **AC←(FS).**

Operandul sursă este încărcat în acumulatorul specificat. Instrucțiunea este exactă și permite utilizarea variabilei nedefinite —0 care este încărcată în acumulator chiar dacă întreruperea corespunzătoare este invalidată; dacă (FS)=—0, atunci ambii indicatori FN și FZ sînt setați la 1. Dacă FIUV=1 și (FS)=—0, atunci apare o întrerupere înainte ca AC să fie încărcat. Nu pot apărea depășiri.

— *poziționarea indicatorilor de condiții:*

FN=1 dacă (AC) rezultat este negativ, altfel FN=0.
FZ=1 dacă (AC) rezultat este nul, altfel FZ=0.
FV=0.
FC=0.

3.4.6.2. **Instrucțiunea LDC(DF/FD)**

— *mnemonică și cod:*

LDCDF LDCFD	177(A+4)FS

— *nume:* load and convert from double to floating/floating to double (încărcare și conversie între precizii).

— *operație:* **AC←C_{xy}(FS),**

unde C_{xy} specifică conversia din modul flotant X în modul flotant Y; X=D și Y=F dacă FD=0 (simplă precizie); X=F și Y=D dacă FD=1 (dublă precizie). Execuția instrucțiunii depinde, deci, de starea indicatorului FD din registrul de stare:

a) dacă modul curent este flotant F (FD=0), atunci operandul sursă este considerat în dublă precizie (pe 4 cuvinte) și este încărcat în acumulatorul specificat AC după ce a fost convertit în simplă precizie (în acest caz mnemonica instrucțiunii este LDCDF); mantisa numărului „scurtat" este trunchiată sau rotunjită, conform stării indicatorului FT. S-ar putea ca în urma operației de rotunjire să apară depășire superioară (overflow); dacă FIV=1 apare întrerupere și rezultatul depășit al conversiei este memorat în AC (el trebuie să fie +0 sau —0); dacă FIV=0 atunci: AC←0 exact.

b) dacă modul curent este dublu D(FD=1), atunci operandul sursă este considerat reprezentat în simplă precizie (pe 2 cuvinte) și este încărcat în partea stîngă a acumulatorului specificat; partea dreaptă, de pondere mică, este ștearsă (în acest caz mnemonica instrucțiunii este LDCFD); instrucțiunea LDCFD este exactă, depășirea superioară nu poate apărea.

Observații variabile pentru ambele cazuri:

1) dacă EXP(FS)=0, atunci: AC←0 exact;

2) depășire inferioară (underflow) nu poate să apară;

3) dacă (FS)=—0 și FIUV=1 atunci apare întrerupere înainte de execuție; indiferent de întrerupere, indicatorii FN și FZ sînt setați în 1.

— *poziționarea indicatorilor de condiții:*

FN=1 dacă (AC)<0, altfel FN=0.
FZ=1 dacă (AC)=0, altfel FZ=0.
FV=1 dacă conversia a produs depășire superioară, altfel FV=0.
FC=0.

3.4.6.3. **Instrucţiunea LDC(IF/ID/LF/LD)**

— *mnemonică şi cod:*	LDCIF LDCID LDCLF LDCLD	177ASS

— *nume:* load and convert integer or long integer in floating or double precision (încărcare şi conversie din întreg în flotant).

— *operaţie:* $AC \leftarrow C_{jx}$ (SS),

unde C_{jx} specifică conversia din modul întreg J în modul flotant X; J=I dacă FL=0 (modul întreg, simplă precizie) sau J=L dacă FL=1 (modul lung, dublă precizie); X=F dacă FD=0 (modul flotant, simplă precizie) sau X=D dacă FD=1 (modul dublu, dublă precizie). În funcţie de starea indicatorilor FL şi FD, instrucţiunea are 4 interpretări:

a) LDCIF dacă FL=0 şi FD=0;
b) LDCID dacă FL=0 şi FD=1;
c) LDCLF dacă FL=1 şi FD=0;
d) LDCLD dacă FL=1 şi FD=1.

Conversia constă în transformarea operandului sursă interpretat ca număr întreg cu precizia J în număr flotant cu precizia X; după conversie, numărul flotant este depus în acumulatorul specificat AC. SS foloseşte modurile de adresare ale procesorului central. Dacă întregul este specificat ca fiind dublă precizie (pe 32 biţi) şi modul de adresare este direct registru (0) sau imediat (modul 2 cu PC), atunci numai 16 biţi (un cuvînt) sînt exploataţi şi anume: cei 16 biţi formează partea semnificativă a numărului (stînga), ceilalţi 16 biţi fiind încărcaţi cu 0 înainte de conversie: LDCIF, LDCID şi LDCLD sînt instrucţiuni exacte; în cazul instrucţiunii LDCLF, partea fracţionară a reprezentării în flotant va fi trunchiată sau rotunjită în flotant, conform indicatorului FT din registrul de stare FPP. Depăşiri nu pot apărea decît în urma unor operaţii de rotunjire (cu constante de rotunjire semnificative). De asemenea, SS nefiind mod de adresare al FPP, nu poate să apară întrerupere la variabila nedefinită —0.

— *poziţionarea indicatorilor de condiţii:*

FN=1 dacă (AC)<0, altfel FN=0.
FZ=1 dacă (AC)=0, altfel FZ=0.
FV=0.
FC=0.

3.4.6.4. **Instrucţiunea LDEXP**

— *mnemonică şi cod:*	LDEXP	176(A+4)SS

— *nume:* load exponent (încărcare exponent).

— *operaţie:* EXP (AC)$\leftarrow$[(SS)+200]$_{0-7}$ (octal).

Biţii 0—7 (octetul inferior) ai sumei dintre operandul sursă şi 200 (octal) sînt încărcaţi în poziţiile 55—62 (locul caracteristicii) din acumulatorul specificat; operandul sursă este interpretat ca număr întreg (dacă e negativ este reprezentat în comple-

ment față de 2); prin adunarea cu 200 (octal) se obține exponentul în exces față de 200 (caracteristica) numărului flotant din AC; restul acumulatorului AC rămîne nemodificat. Cazurile de excepție sînt următoarele:

1) dacă (SS) > 177 (octal) rezultatul sumei cu 200 este minim 400 și va fi interpretat ca depășire superioară (overflow); dacă FIV=1 se va genera o întrerupere și doar biții 0—7 ai sumei vor fi depuși în AC; dacă FIV=0 nu apare întrerupere și AC←0 exact.

2) dacă (SS) < —177 (octal), rezultatul sumei cu 200 este maximum 0 și va fi interpretat ca depășire inferioară (underflow); dacă FIU=1 se va genera o întrerupere și biții 0—7 ai sumei vor fi depuși în AC; dacă FIV=0 nu apare întrerupere și AC←0 exact.

3) nu se generează întrerupere dacă —0 apare în AC, chiar dacă FIUV=1.

— *poziționare indicatori de condiții:*
FN=1 dacă (AC) <, altfel FN=0.
FZ=1 dacă EXP(AC)=0, altfel FZ=0.
FV=1 dacă (SS) > 177 (octal), altfel FV=0.
FC=0.

3.4.6.5. **Instrucțiunea ST(F/D)**

— *mnemonică și cod:*	STF STD	174AFD

— *nume:* store floating/double (memorare mod flotant/dublu).
— *operație:* **FD←(AC).**

Se depune conținutul acumulatorului specificat la adresa destinație (în memoria centrală sau într-un alt acumulator dacă modul de adresare este 0) pe două cuvinte sau pe patru cuvinte, conform preciziei indicate de FD. În cazul simplei precizii (FD=0) este memorată partea stîngă (mai semnificativă) a acumulatorului; în cazul modului de adresare imediat (2 cu PC) se memorează după instrucțiune un singur cuvînt (cei 16 biți mai semnificativi din AC), indiferent de precizie. Instrucțiunea este exactă; nu pot apărea depășiri. Variabila nedefinită —0 nu produce întrerupere, indiferent de starea indicatorului FIUV, deoarece ea se află în acumulator și nu în memorie; această instrucțiune permite deci memorarea lui —0 aflat într-un acumulator din FPP. Totuși, FPP nu poate memora —0 în AC decît dacă apare în conjuncție cu o depășire flotantă superioară sau inferioară și întreruperea respectivă este validată; în acest fel, utilizatorul are posibilitatea să șteargă o variabilă nedefinită transferînd-o din acumulatorul în care se află în el însuși.

— *indicatorii de condiții* nu sînt afectați.

3.4.6.6. **Instrucțiunea STC(FD/DF)**

— *mnemonică și cod:*	STCFD STCDF	176AFD

— *nume:* store and convert from floating to double/double to floating (memorare și conversie între precizii).

— *operație:* **FD←C_{xy}(AC),**

unde Cxy reprezintă conversia din modul flotant X în modul flotant Y; X=F și Y=D dacă FD=0 (simplă precizie); X=D și Y=F dacă FD=1 (dublă precizie). Execuția instrucțiunii (cît și mnemonica ei) depinde de starea indicatorului FD din registrul de stare al FPP:

a) dacă modul curent este flotant (FD=0), atunci conţinutul acumulatorului specificat este memorat la destinaţie (pe patru cuvinte), cele 2 cuvinte mai puţin semnificative (partea dreaptă) ale destinaţiei fiind şterse; instrucţiunea este STCFD (memorare şi conversie din simplă în dublă precizie). Instrucţiunea STCFD este exactă, depăşire superioară nu poate apărea.

b) dacă modul curent este dublu (FD=1), atunci conţinutul acumulatorului specificat este convertit în simplă precizie şi depus la destinaţie (pe 2 cuvinte); instrucţiunea este STCDF (memorare şi conversie din dublă în simplă precizie). Conversia se face prin trunchierea sau rotunjirea mantisei „scurtate", conform stării bitului FT din registrul de stare al FPP. În urma operaţiei de rotunjire poate apărea depăşire flotantă superioară (overflow); dacă FIV=1, apare întrerupere şi la destinaţie se memorează rezultatul depăşit (care trebuie să fie +0 sau —0); dacă FIV=0, nu apare întrerupere şi FD←0 exact.

Observaţii valabile în ambele cazuri:

1) dacă EXP(AC)=0, atunci FD←0 exact;

2) depăşire inferioară (underflow) nu poate apărea;

3) variabila nedefinită —0 nu produce întrerupere, chiar dacă FIUV=1, deoarece operandul sursă se află în acumulatorul din FPP.

— *poziţionarea indicatorilor de condiţii:*

FN=1 dacă (AC)<0, altfel FN=0.
FZ=1 dacă (AC)=0, altfel FZ=0.
FV=1 dacă apare depăşire superioară la conversie, altfel FV=0.
FC=0.

3.4.6.7. Instrucţiunea STC(FI/FL/DI/DL)

— *mnemonică şi cod:*

STCFI STCFL STCDI STCDL	175(A+4)DD

— *nume:* store and convert from floating or double to integer or long integer (memorare şi conversie din flotant în întreg).

— *operaţie:* **DD←Cxj(AC),**

unde Cxj specifică conversia din modul flotant X în modul întreg J; X=F în simplă precizie flotantă (FD=0) sau X=D în dublă precizie flotantă (FD=1); J=I în simplă precizie întreagă (FL=0) sau J=L în dublă precizie întreagă (FL=1). În funcţie de starea indicatorilor FD şi FL din registrul de stare al FPP, instrucţiunea are 4 interpretări:

a) STCFI dacă FD=0 şi FL=0;
b) STCFL dacă FD=0 şi FL=1;
c) STCDI dacă FD=1 şi FL=0;
d) STCDL dacă FD=1 şi FL=1.

Conversia constă în transformarea operandului sursă din acumulatorul specificat AC într-un număr întreg (dacă e negativ va fi reprezentat în complement faţă de 2). În cazul simplei precizii flotante (FD=0), numărul flotant este trunchiat (indiferent de starea bitului FT din registrul de stare al FPP), considerîndu-se doar jumătatea stîngă (de pondere superioară) din AC. În urma conversiei se poate depăşi capacitatea de reprezentare a întregului rezultat (N); N trebuie să fie cuprins între limitele:

$-2^{15} \leqslant N \leqslant 2^{15-1}$ pentru FL=0 (simplă precizie) sau
$-2^{31} \leqslant N \leqslant 2^{31-1}$ pentru FL=1 (dublă precizie).

Limitele acestea sînt impuse de lungimea destinaţiei în care urmează să fie memorat întregul N şi anume: dacă FL=0 la destinaţie se va memora un cuvînt (16 biţi);

dacă FL=1 la destinație se vor memora două cuvinte (32 biți). Excepție fac cazurile în care FL=1 și modul de adresare este 0 (direct registru din procesorul central) sau imediat (modul 2 cu PC): imediat după instrucțiune sau în registrul general R se vor memora numai 16 biți mai semnificativi ai întregului N. Dacă este depășită capacitatea de memorare a întregului N apare eroarea de conversie flotant — întreg și conținutul destinației este forțat la 0: DD←0; indicatorul de condiție FC este setat la 1 și dacă FIC=1, procesorul de virgulă mobilă va genera o întrerupere. Nu apare întrerupere la variabila nedefinită —0 (chiar dacă FIUV=1) deoarece sursa este în acumulatorul din FPP și nu în memorie.

— *poziționarea indicatorilor de condiții:*

FN=1 dacă (DD)<0, altfel FN=0.
FZ=1 dacă (DD)=0, altfel FZ=0.
FV=0.
FC=1 dacă apare eroare de conversie (depășirea capacității de reprezentare a numărului întreg), altfel FC=0.

Acești indicatori, astfel poziționați, sînt memorați și în indicatorii corespunzători din procesorul central (PSW):

N←FN; Z←FZ; V←FV și C←FC.

3.4.6.8. **Instrucțiunea STEXP**

— *mnemonică și cod:* | STEXP 175ADD |

— *nume:* store exponent (memorare exponent).
— *operație:* **DD←EXP(AC)—200 (octal).**

Din caracteristica numărului flotant aflat în acumulatorul specificat se scade 200 (octal) și rezultatul este memorat la destinație; în acest mod, la destinație se obține exponentul real al numărului reprezentat în acumulator (dacă e negativ, va fi reprezentat în complement față de 2). Instrucțiunea este exactă. Nu pot apărea depășiri. Dacă în acumulator se găsește variabila nedefinită —0 nu apare întrerupere (chiar dacă FIUV=1).

— *poziționarea indicatorilor de condiții:*

FN=1 dacă (DD)<0, altfel FN=0.
FZ=1 dacă (DD)=0, altfel FZ=0.
FV=0.
FC=0.

Acești indicatori sînt transferați în indicatorii corespunzători din procesorul central (PSW):

N←FN; Z←FZ; V←FV și C←FC.

3.4.7. INSTRUCȚIUNI DE STARE

Instrucțiunile prezentate în continuare permit accesul utilizatorului la registrele de stare și de erori ale FPP. Ele nu operează cu numere flotante sau întregi, ci cu date binare de stare, deci nu ridică probleme de precizie; de asemenea, nu poate apărea nici o eroare și nici o întrerupere generată de FPP. Indicatorii de condiții din FPP nu sînt modificați în urma execuției acestor instrucțiuni (cu excepția LDFPS).

3.4.7.1. **Instrucțiunea CFCC**

— *mnemonică și cod:* | CFCC 170000 |

— *nume:* copy floating condition codes (transfer indicatori de condiţii flotant).
— *operaţie:* N←FN.
Z←FZ.
V←FV.
C←FC.

Indicatorii de condiţii din registrul de stare a FPP sînt transferaţi în indicatorii corespunzători din registrul de stare program (PSW) al procesorului central.

3.4.7.2. **Instrucţiunea SETF**

— *mnemonică şi cod:*

SETF	170001

— *nume:* set floating mode (set mod flotant F).
— *operaţie:* FD←0.
Este şters indicatorul FD din registrul de stare al FPP, ulterior urmînd să se interpreteze numerele flotante utilizate reprezentate în simplă precizie (modul flotant F).

3.4.7.3. **Instrucţiunea SETD**

— *mnemonică şi cod:*

SETD	170011

— *nume:* set floating double mode (set mod flotant dublu D).
— *operaţie:* FD←1.
Indicatorul FD din registrul de stare al FPP este setat în 1; numerele flotante utilizate ulterior vor fi considerate reprezentate în dublă precizie (modul dublu D)

3.4.7.4. **Instrucţiunea SETI**

— *mnemonică şi cod:*

SETI	170002

— *nume:* set integer mode (set mod întreg I).
— *operaţie:* FL←0.
Indicatorul FL din registrul de stare al FPP este şters; numerele întregi utilizate ulterior vor fi considerate reprezentate în simplă precizie (modul întreg I).

3.4.7.5. **Instrucţiunea SETL**

— *mnemonică şi cod:*

SETL	170012

— *nume:* set long integer mode (set mod întreg lung L).
— *operaţie:* FL←1.
Indicatorul FL din registrul de stare al FPP este setat la 1; numerele întregi utilizate ulterior vor fi considerate reprezentate în dublă precizie (modul lung L).

3.4.7.6. **Instrucţiunea LDFPS**

— *mnemonică şi cod:*

LDFPS	1701SS

— *nume:* load FPP's program status (încărcare registru de stare al FPP).
— *operație:* **FPS←(SS).**
Operandul sursă (un cuvînt) este încărcat în registrul de stare FPS al procesorului de virgulă mobilă. Instrucțiunea este utilizată pentru a valida sau invalida anumite întreruperi pentru a poziționa convenabil indicatorii FD, FL și FT sau pentru a șterge indicatorul de eroare globală FER. Indicatorii de condiții sînt poziționați conform valorilor binare 0—3 din operandul sursă.

3.4.7.7. Instrucțiunea STFPS

— *mnemonică și cod:*

STFPS	1703DD

— *nume:* store FPP's program status (memorare registru de stare al FPP).
— *operație:* **DD←(FPS).**
Conținutul registrului de stare FPS din procesorul de virgulă mobilă este memorat la destinația DD (calculată conform modurilor de adresare ale procesorului central).

3.4.7.8. Instrucțiunea STST

mnemonică și cod:

STST	1703DD

— *nume:* store FPP's status (memorare registre de eroare ale FPP).
— *operație:* **DD←(FEC)**
DD+2←(FEA).
La adresa indicată de destinația DD este memorat registrul FEC (aceasta conține codul erorii detectate de FPP); la adresa următoare este memorat registrul FEA (care conține adresa logică a instrucțiunii care a provocat eroarea). În cazul modului de adresare direct-registru general (0) sau a modului de adresare imediat (2 cu PC), numai registrul FEC este salvat. Datele din FEC și FEA sînt actuale numai dacă ultima instrucțiune de virgulă flotantă executată a provocat apariția unei erori.

3.4.8. SCRIEREA SINTACTICĂ A INSTRUCȚIUNILOR DE VIRGULĂ MOBILĂ

Pentru scrierea instrucțiunilor de virgulă mobilă se aplică aceleași criterii ca și la instrucțiunile procesorului central, prezentate în capitolul 2. Sînt utilizate mnemonicele instrucțiunilor, precum și mnemonicele modurilor de adresare (acestea din urmă fiind aceleași cu cele folosite pentru procesorul central, vezi 2.1.4.). În cazul instrucțiunilor cu doi operanzi, operanzii sînt separați prin virgulă, dar ordinea lor nu coincide cu cea din codul instrucțiunii, ci cu ordinea firească indicată de operația executată (sursă, destinație):

a) în cazul instrucțiunilor de încărcare (cu sau fără conversie) sau a operațiilor aritmetice, mai întîi se scrie operandul sursă (codificat în biții 0—5 ai instrucțiunii) și apoi acumulatorul folosit (în care se obține rezultatul operației);

b) în cazul instrucțiunilor de memorare, mai întîi se scrie acumulatorul folosit (operandul sursă) și apoi destinația care poate fi plasată oriunde.

În exemplul următor este redat un program de adunare a N numere reprezentate în flotant în dublă precizie (deci, anterior, vor trebui poziționați indicatorii corespunzători din registrul de stare al FPP: FD=1 și FL=1). Numerele se află în memoria centrală, pe cîte 4 cuvinte fiecare, începînd cu adresa AD.NR. Primele două instrucțiuni realizează încărcarea adresei AD.NR. în R0 (folosit ca pointer pentru stiva numerelor) și a numărului N în R1. Suma se calculează în acumulatorul AC0 (în prealabil acesta este șters); după fiecare adunare (ADDD) pointerul R4 este autoincrementat cu 10 (octal), adresînd următorul număr. Adunarea se repetă pînă ce

numărul N este epuizat, folosindu-se decontorizarea cu instrucţiunea SOB. Penultimele două instrucţiuni realizează depunerea rezultatului în memorie:

— rezultatul în flotant este memorat (STD) imediat după numerele adunate (pe 4 cuvinte);

— rezultatul convertit în număr întreg este depus la adresele următoare, pe două cuvinte (cu instrucţiunea STCDL=memorează şi converteşte din flotant în întreg dublă precizie). Instrucţiunea HALT va afişa la panou conţinutul registrului general R0, deci adresa rezultatului întreg (aplicaţie: AD.NR.=10000 şi N=5).

```
1000 012700 010000    A0    MOV     #AD.NR., R0
1004 012701 000005          MOV     #N, R1
1010 170400                 CLRD    AC0
1012 172020           A1    ADDD    (R0)+, AC0
1014 077102                 SOB     R1, A1
1016 174020                 STD     AC0, (R0)+
1020 175410                 STCDL   AC0, (R0)
1022 000000                 HALT
```

Observaţie: Instrucţiunile CLRD, ADDD, STD şi STCDL acţionează procesorul de virgulă mobilă FPP; ele nu se pot executa decît dacă acest procesor există în configuraţia sistemului (în caz contrar, codurile respective vor fi interpretate ca rezervate şi se va semnala eroarea de instrucţiune inexistentă).

ANEXA 1

Lista codurilor ASCII

Octal Code	Char	Octal Code	Char	Octal Code	Char	Octal Code	Char
000	NUL	040	SP	100	@	140	′
001	SOH	041	!	101	A	141	a
002	STX	042	″	102	B	142	b
003	ETX	043	#	103	C	143	c
004	EOT	044	$	104	D	144	d
005	ENQ	045	%	105	E	145	e
006	ACK	046	&	106	F	146	f
007	BEL	047	′	107	G	147	g
010	BS	050	(	110	H	150	h
011	HT	051	)	111	I	151	i
012	LF	052	*	112	J	152	j
013	VT	053	+	113	K	153	k
014	FF	054	’	114	L	154	l
015	CR	055	—	115	M	155	m
016	SO	056	.	116	N	156	n
017	SI	057	/	117	O	157	o
020	DLE	060	0	120	P	160	p
021	DC1	061	1	121	Q	161	q
022	DC2	062	2	122	R	162	r
023	DC3	063	3	123	S	163	s
024	DC4	064	4	124	T	164	t
025	NAK	065	5	125	U	165	u
026	SYN	066	6	126	V	166	v
027	ETB	067	7	127	W	167	w
030	CAN	070	8	130	X	170	x
031	EM	071	9	131	Y	171	y
032	SUB	072	:	132	Z	172	z
033	ESC	073	;	133	[	173	{
034	FS	074	<	134	\	174	|
035	GS	075	=	135	]	175	}
036	RS	076	>	136	Λ	176	⌒
037	US	077	?	137	—	177	DEL

ANEXA 2

Lista vectorilor de întrerupere

VECTOR (octal)	
000	rezervat
004	erori procesor central (instrucțiuni ilegale, erori de adresare impară, erori de bus, depășire stivă)
010	instrucțiuni rezervate
014	instrucțiunea BPT sau poziționare bit T din PSW
020	instrucțiunea IOT
024	cădere tensiune
030	instrucțiunea EMT
034	instrucțiunea TRAP
040 — 054	rezervate pentru sistemul software
060	consolă (cititor)
064	consolă (imprimator)
070	cititor bandă de hîrtie
074	perforator bandă de hîrtie
100	ceas de linie
104	ceas timp real programabil
110	—
114	eroare paritate la memoria centrală
120	XY Plotter
124 — 174	rezervate sau utilizate de unități periferice speciale
160	disc de masă (DD — INDEPENDENT)
200	imprimantă paralelă (LP11)
204	disc capete fixe (RF11)
210	disc RC11
214	DECtape (TC11)
220	disc cortuș (cartridge — RK11)
224	bandă magnetică (TM11)
230	cititor cartele (CR11)
234 — 240	utilizări speciale
244	procesor virgulă flotantă
250	management memorie
254	disc de masă (RP11 — CORAL)
260	casetă magnetică (TA11)
264	disc flexibil (floppy — RX11)
270 — 274	rezervate
300 — 776	teletransmisie (cuploare sau multiplexoare sincrone sau asincrone).

ANEXA 3

Lista adreselor de serviciu

760010 — 763776	—	multiplexoare asincrone şi cuploare sincrone
765000 — 765776	—	a doua jumătate a emulatorului de consolă
767760 — 767774	—	interfeţe paralele (CORAL)
770000 — 770006	—	panou comandă I-100
772110 — 772136	—	controloare paritate pentru memoria centrală
772300 — 772316	—	registrele PDR pentru spaţiul de instrucţiuni KERNEL
772320 — 772336	—	registrele PDR pentru spaţiul de date KERNEL
772340 — 772356	—	registrele PAR pt. spaţiul de instrucţiuni KERNEL
772360 — 772376	—	registrele PAR pt. spaţiul de date KERNEL
772440 — 772476	—	banda magnetică
772516	—	registrul de stare 3 management (SR3)
772540 — 772544	—	ceas timp real programabil
773000 — 773766	—	prima jumătate a emulatorului de consolă
775610 — 776176	—	cuploare asincrone cu o cale (pt. terminale de distanţă)
776500 — 776676	—	cuploare asincrone cu o cale (pt. terminale apropiate)
776710 — 776736	—	cuplor disc de masă (CORAL)
777160 — 777164	—	cititor cartele
777170 — 777172	—	disc flexibil (CORAL)
777220 — 777236	—	cuplor disc de masă (INDEPENDENT)
777400 — 777416	—	cuplor disc cartuş
777500 — 777502	—	casetă magnetică
777514 — 777516	—	imprimantă paralelă
777546	—	ceasul de linie
777550 — 777552	—	cititor bandă de hîrtie
777554 — 777556	—	perforator bandă de hîrtie
777560 — 777566	—	consola operator
777570	—	consola de switch-uri şi afişare (CORAL)
777572	—	registrul de stare 0 management (SR0)
777576	—	registrul de stare 2 management (SR2)
777600 — 777616	—	registrele PDR pt. spaţiul de instrucţiuni USER
777620 — 777636	—	registrele PDR pt. spaţiul de date USER
777640 — 777656	—	registrele PAR pt. spaţiul de instrucţiuni USER
777660 — 777676	—	registrele PAR pt. spaţiul de date USER
777744 — 777746	—	registrele memoriei cache
777776	—	registrul de stare program (PSW).

ANEXA 4

Lista instrucțiunilor în ordinea codurilor

OP Code	Mnemonic
000000	HALT
000001	WAIT
000002	RTI
000003	BPT
000004	IOT
000005	RESET
000006	RTT
000007 : 000077	neutilizate
0001DD	JMP
00020R	RTS
000210 : 000227	neutilizate
00023N	SPL
000240	NOP
000241 : 000277	poziționare indicatori condiții
0003DD	SWAB
0004XXX	BR
0010XXX	BNE
0014XXX	BEQ
0020XXX	BGE
0024XXX	BLT
0030XXX	BGT
0034XXX	BLE
004RDD	JSR
0050DD	CLR
0051DD	COM
0052DD	INC
0053DD	DEC
0054DD	NEG
0055DD	ADC

OP Code	Mnemonic
0060DD	ROR
0061DD	ROL
0062DD	ASR
0063DD	ASL
0064NN	MARK
0065SS	MFPI
0066DD	MTPI
0067DD	SXT
007000 : 007777	neutilizate
01SSDD	MOV
02SSDD	CMP
03SSDD	BIT
04SSDD	BIC
05SSDD	BIS
06SSDD	ADD
070RSS	MUL
071RSS	DIV
072RSS	ASH
073RSS	ASHC
074RDD	XOR
07500R	FADD
07501R	FSUB
07502R	FMUL
07503R	FDIV
075040 : 076777	neutilizate
077RNN	SOB
1000XXX	BPL
1004XXX	BMI
1010XXX	BHI
1014XXX	BLOS
1020XXX	BVC
1024XXX	BVS
1030XXX	BCC; BHIS
1034XXX	BCS; BLO

OP Code	Mnemonic
104000 : 104377	EMT
104400 : 104777	TRAP
1050DD	CLRB
1051DD	COMB
1052DD	INCB
1053DD	DECB
1054DD	NEGB
1055DD	ADCB
1056DD	SBCB
1057DD	TSTB
1060DD	RORB
1061DD	ROLB
1062DD	ASRB
1063DD	ASLB
1064SS	MTPS
1065SS	MFPD
1066DD	MTPD
1067DD	MFPS
107000 : 107777	neutilizate
11SSDD	MOVB
12SSDD	CMPB
13SSDD	BITB
14SSDD	BICB
15SSDD	BISB
16SSDD	SUB
170000 : 177777	instrucțiuni FPP

CONSFĂTUIREA NAȚIONALĂ DE CERCETARE-PROIECTARE ASISTATĂ DE CALCULATOR CPAC '84

dr. ing. G. Manolescu
I.C.T.I.

Prima consfătuire națională de cercetare-proiectare asistată de calculator, CPAC'84, organizată sub egida CNST de către ICTCM, ICPE, ISPE, ICEMENERG, IPCT și ICI, s-a desfășurat în perioada iunie—iulie 1984 și a avut drept obiectiv comunicarea rezultatelor deosebite, teoretice și practice, obținute în domeniu, facilitarea unui larg schimb de opinii între specialiști și conturarea stadiului actual al realizărilor.

Lucrările s-au desfășurat pe secțiuni conform programului alăturat.

O privire de ansamblu asupra lucrărilor consfătuirii, scoate în evidență următoarele aspecte:

● Lucrările prezentate au arătat că, în comparație cu perioada 1980—1981, care a constituit momentul demarării unor preocupări susținute în domeniu, stadiul actual marchează o maturizare atît în plan practic, cît și în plan teoretic.

● Numărul lucrărilor CPAC intrate în exploatare curentă, precum și numărul beneficiarilor acestor lucrări, demonstrează începutul unei informatizări de masă a cercetării-proiectării.

● Efectele economice, de multe ori spectaculoase, au trecut din sfera „potențialului" în cea a „realității".

● Realizările semnificative obținute prin utilizarea calculatorului în diversele faze ale ciclului de viață al produselor tehnice: marketing, concepție, proiectare, fabricație, impun realizarea unor sisteme integrate.

● Necesitatea, subliniată în luările de cuvînt ale multor participanți, ca manifestarea de genul CPAC'84 să devină periodică (s-a propus ca o asemenea consfătuire să aibă loc o dată la 2 ani).

● S-a constatat că, în perioada imediat următoare, direcțiile de interes major în domeniul CPAC sînt:

— grafica pe calculator;
— interfețe de dilog „om-calculator";
— prelucrarea datelor experimentale;
— prelucrarea datelor provenite din teledetecție (inclusiv prin discretizare de imagini);
— ingineria structurală (cu utilizarea, îndeosebi, a metodelor elementului finit și a elementului de frontieră);
— instrumente/utilitare pentru realizarea performantă și cu productivitate sporită a aplicațiilor CPAC pe diverse domenii.

Programul consfătuirii este dat în continuare, urmat de ciclul „Proiectare asistată de calculator".

1. MEODE TEHNICI ȘI INSTRUMENTE PENTRU REALIZAREA APLICAȚIILOR CPAC

Organizator: CNST-ICI

1. **Priorități în CPAC Dr. ing. M. Guran — ICI**
2. **Concepția centrului de calcul privind automatizarea lucrărilor de inginerie tehnologică în sistemele MILMC** dr. ing. M. Mureșan, ing. V. Georgescu, ing. V. Paraschiv, mat. D. Constantin, ing. L. Enăchescu — Centrul de calcul MILMC.

3. **Necesitatea realizării unui mediu de programe dedicat CPAC** ing. O. Stănescu — CNST.
4. **Un sistem prototip CPAC** dr. ing. G. Manolescu — I.C.I.
5. **O tehnică de specificare a interfeţelor de dialog „om-calculator" cu aplicare în CPAC** mat. R. Constantinescu, mat. A. Lepădatu, mat. C. Lepădatu, dr. ing. G. Manolescu, ec. A. Tudor.
6. **Interfaţa cu utilizatorul: concepte şi utilizare practică în produsele program pentru cercetarea-proiectarea asistată de calculator** I. Roman, A. Toia, R. Catarahia, S. Crăciunaş — ICEM.
7. **Consideraţii asupra interfeţei om-calculator pentru conducerea proceselor într-o centrală nucleară** ing. A. Tănăsescu, ing. I. Turcu IRNE — Piteşti.
8. **Consideraţii asupra unui generator de interfeţe om-calculator** mat. A. Lepădatu — ICI.
9. **Sisteme consultant, consilier şi specialist; produsul conexiuni** L. Dragomirescu, Inst. V. Babeş.
10. **Compararea A P eşantioane primitive de „sistem consultant" aplicaţii** L. Dragomirescu Inst. V. Babeş.
11. **CASAD — pachet interactiv pentru analiza şi proiectarea asistată de calculator a sistemelor automate** dr. ing. A. Davidoviciu, dr. ing. A. Varga — I.C.I.
12. **Metode pentru stimularea creativităţii — orientare majoră în activitatea CPAC** A. Toia, ICEM.
13. **Modele în limbaje de simulare realizate la CCUB, pentru aplicaţii în industria constructoare de maşini, industria chimică şi agricultură** Conf. dr. I. Văduva, M. Lovin, M. Bogdan, D. Panaite, CCUB.
14. **SIM — mediu interactiv de simulare a sistemelor dinamice** dr. A. M. Necula, Dr. E. Kalisz, IPB.
15. **Utilizarea metodei Monte-Carlo în probleme de optimizare cu restricţii liniare** mat. S. Ştefănescu, CCUB.
16. **Industria asistată de calculator** M. Râşnoveanu, IPA.
17. **Metode rapide pentru generarea cu calculatorul a vectorilor aleatori** Conf. dr. I. Văduva, C.C.U.B.
18. **Pachet de programe utilizat la prelucrarea statistică a chestionarelor** mat. P. Ştefănescu, mat. S. Ştefănescu, CCUB.
19. **Procedeu de extensie dinamică a tablourilor în fortran pentru sistemul FELIX** ing. G. Dumitrescu, mat. C-tin Ursu, CTCE — Piatra Neamţ.
20. **Sistem de descriere, manipulare şi reprezentare a ansamblurilor compuse din corpuri poliedrale** ing. Cornel Costea, mat. Ioan Moldovan, mat. Dorin Pănut, dr. ing. Elinor Tăciulescu, CTCE Cluj-Napoca.
21. **Produs-program pentru realizarea interactivă a variantelor de desen utilizînd terminalele de tip display grafic sau semigrafic** ing. C-tin Jeican, mat. Valer Anisiu, ing. Lucian Rusu, mat. Anca Tămaş, CTCE Cluj-Napoca.
22. **Sistem de grafică bazat pe dialog om-calculator** ec. A. Tudor — ICI.
23. **Îmbunătăţiri ale sistemului de operare ARISTO** prof. ing. I. G. Carabogdan, ing. A. Ionescu, ing. F. Roşu, IPB.
24. **SAGAS (sintaxe aritmetice cu generare automată de şiruri) o metodă eficientă de proiectare a aplicaţiilor CPAC** mat. M. Olaru, ec. T. Boagiu, ec. D. Roman, aj. an. pr. G. Pascu — ICPE.
25. **DAC-84 (desenarea asistată de calculator) — limbaj de desenare realizat pe structuri SAGAS** mat. Mircea Olaru, mat. Cristina Iftode, mat. Adelina Stoian, ec. Mariana Cegăneanu ICPE.
26. **Pachet de programare pentru reprezentări în spaţiu** mat. Mircea Olaru, mat. Viorel Lungu, aj. an. Alexandra Crăciunescu ICPE.
27. **MELFIT — 2D, 3D — metoda elementului finit în 2D şi 3D — pachet de programe pentru rezolvarea ecuaţiilor cu derivate parţiale în probleme de electrotehnică (sau echivalente)** mat. M. Olaru, mat. C. Iftode, ec. T. Boagiu, ICPE.
28. **Componente pentru memorarea şi manipularea desenelor plane şi în spaţiu** ing. S. Vlasiu, ASG.
29. **Limbaj pentru descrierea şi manipularea formelor geometrice** dr. ing. A. Ioachim, mat. V. Firta, mat. G. Popa — ASG.
30. **Intergraf — produs program pentru reprezentări grafice, în mod interactiv, pe**

microcalculatorul M118 mat. C. Lepădatu — I.C.I., ec. T. Şurubaru — CTCE Ploieşti.

31. **Sistem de achiziţie şi prelucrare automată, în timp real, a vorbirii, specializat pentru recunoaşterea formelor de tip vocalic** V. Groza, L. Fortuna, I. Gândoiu, I. P. „Traian Vuia" Timişoara.
32. **Corelarea metodelor de conversie analog numerică cu metodele de analiza parametrică a vorbirii, în scopul reducerii debitului informaţional binar rezultat — cu aplicaţii în realizarea sistemelor cu inteligenţa artificială** L. Fortuna, I. P. „Traian Vuia" Timişoara.
33. **Vocoder de tip spectral, utilizat ca terminal de intrare/ieşire pentru realizarea dialogului direct om-calculator în limbaj natural vorbit** L. Fortuna, O. Racz, I. P. „Train Vuia" Timişoara.
34. **Sistem de recunoaşterea ordinelor vorbite fără formarea preliminară a etaloanelor contrapuse („sistem olimpic de adaptarea deciziei")** I. Bogoş, ISRB — Bucureşti.
35. **Trasarea reţelelor kilometrice proprii imaginilor LANSDAT clasificate** dr. ing. N. Zegheru, ing. M. D. Anton, I.G.F.C.O.T. — Bucureşti.
36. **Prelucrarea datelor spectrometrice de masă transmise de pe rachetele geofizice vertical** C. Jalobeanu, M. Jalobeanu, D. Ristoiu, A. Romanţan, I.T.I.M. Cluj-Napoca.
37. **Un sistem de prelucrarea datelor digitale de teledetecţie** ing. M. Albotă, fiz. G. Ioanid I.G.F.C.O.T.
38. **BARGRAP — modul SOFTWARE pentru pachete de programe cercetare-proiectare asistată de calculator în energia nucleară** ing. D. Niţu, ing. M. Orhei, ing. V. Orhei, I.R.N.E. — Piteşti.

2. PRODUSE PROGRAM APLICATIVE CPAC ÎN CONSTRUCŢII DE MAŞINI

Organizator: ICCM-ICTCM

1. **Cuvînt de deschidere** ing. I. Crişan, Director general ICCM.
2. **Analiza structurii cercetării şi proiectării asistate de calculator şi a aspectelor tehnice şi economice ale aplicării sale cu referire la proiectarea de avioane, motoare termice şi automobile** dr. ing. S. Săndulescu INMT.
3. **Proiectarea caroseriei de autoturism pentru instalaţia IDS-3 (GERBER)** ing. M. Vartolaş, ing. F. Dan, ing. A. Szakacs, ing. M. Proca, ing. C. Pîlsănescu, ing. A. Antohie, CCSIT-A Piteşti.
4. **Utilizarea programelor de calcul automat în cadrul proiectării unei maşini de haldat** E. Cuteanu, I. Appeltauer, I. Munteanu, M. Iosip, I. P. „Train Vuia" Timişoara, catedra de construcţii metalice.
5. **Folosirea grupelor minimale în modelarea automată a roboţilor industriali,** Gh. Drăgănoiu, E. Oprea, S. Baubec, — I. Automatica; A. Moangă — ICI.
6. **Proiectarea automată a sculelor în construcţia de maşini folosind limbajul LIPCON** ing. Ghürtler Paul, ing. Creţoiu Ştefania ICTCM.
7. **Proiectarea asistată de calculator a maşinilor de cîrmă pentru nave maritime** ing. Bărzoi C-tin, mat. Mîrmea V. CCSIT-UCD Brăila.
8. **Proiectarea optimă asistată de calculator a mecanismelor de suspensie şi direcţie la autoturisme de mic litraj model matematic** cercet. şt. L. Popescu, ing. S. Manoliu, c.p. G. Fotin, mat. V. Ionescu INMT.
9. **Aspecte ale modelării geometrice pentru piese din ramura construcţiilor de maşini** I. Radoslovescu — ICI.
10. **Arhitectura informaţională a sistemelor SEFA-DISROM de proiectare asistată de calculator a dispozitivelor pentru prelucrări pe maşini-unelte** conf. dr. ing. I. Brăgaru — I.P.B.
11. **Captatoare solare de încălzit aer — cercetări asistate de calculator** ing. D. Ionescu, ing. P. Zaru, ICSIT-EE.
12. **Proiectarea asistată de calculator a reductoarelor cilindrice cu axe paralele cu 1, 2 şi 3 trepte** ing. I. Popescu, ing. I. Turcu, mat. B. Porţeanu, mat. P. Brumariu, ing. S. Creţoiu, ICTCM.

13. **Biblioteca de programe pentru analiza structurală — stadiul actual, cerințe și posibilități de dezvoltare** dr. ing. Stere marcel, INCREST.
14. **Sistem interactiv de programe pentru descrierea, calculul și postprelucrarea prin metoda elementului finit, a structurilor,** ing. Ghenu Șerban, ing. Sebe Gh., ing. Stoenescu Florin, ing Tothezan Puiu, INMT.
15. **Geometrizarea și frezarea suprafețelor cu dublă curbură cu ajutorul elementului finit** dr. ing. Berar Cristian — INCREST.
16. **Geometrizarea și frezarea suprafețelor cu dublă curbură cu ajutorul elementului finit de suprafață** mat. Vătui Lan — INCREST.
17. **Analiza stării de tensiuni mecanice la paletele și discul I al Turbinei C.K.A. — 12/10 334 și stabilirea unei soluții de îmbunătățire** ing. M. Ștefănescu, dr. ing. A. Nahlik, ing. N. Dinu, mat. A. Pastia, ICSIT-EE.
18. **Calculul de rezistență al discurilor rotorice și al picioarelor de palete pentru turbine navale** ing. M. Ștefănescu, ing. I. Cornoiu, mat. A. Pastia, ICSIT-EE.
19. **Calculul flanșelor prin metoda elementului finit** ing. D. Lazăr — ICPIAF Cluj.
20. **Realizări în pregătirea tehnologică a fabricației asistată de calculator în industria constructoare de mașini** ing. Dobre Nicolae, director tehnic — ICTCM.
21. **Programarea calculelor de proiectare ale coturilor de aspirație din beton pentru turbine hidraulice** mat. M. Vertan, mat. P. Românu, dr. ing. Gh. Vertan CCSIT-EH Reșița.
22. **Ridicarea diagramei caracteristce a compresoarelor frigorifice cu ajutorul calculatorului** c.s. M. Berinde, ing. M. Damian, ICPIAF Cluj.
23. **Model matematic pentru calculul vitezei de migrație teoretice, în vederea determinării performanțelor electrofiltrelor** mat. D. Mihai, ing. R. Macarie, ICSIT-EE.
24. **Concepția unui sistem de proiectare aisistată de calculator pentru elaborarea interactivă a tehnologiilor pentru piese mecanice prelucrate prin așchiere** dr. ing. C-tin Sîmbotin, ing. D. Ionescu, mat. I. Radoslovescu, mat. M. Moise — ICI.
25. **Modul pentru stabilirea operațiilor tehnologice într-un sistem de proiectare asistată** ing. L. Fărcășanu, mat. I. Radoslovescu — ICI, ing. S. Ionescu — IPB.
26. **Generarea suprafețelor complexe în vederea prelucrării pe MUCN, cu ajutorul limbajului LIPCON-S** mat. Vaida Theodor — ICTCM, mat. Gheorghe Paul — ICTCM.
27. **Sistem integrat cu minicalculator pentru rezolvarea problemelor tehnologice de execuție și control pe MUCN** ing. Georgescu Florin, ing. Savu Mircea, ing. Nicola Ilie, ing. Cobzaru Alex., ing. Șerbănescu Radu — INMT.
28. **Utilizarea comenzii numerice în execuția modelelor de caroserie** ing. Proca M., ing. Fîlsănescu, ing. Antohie, ing. Vertolaș Marcel, ing. Dan Florian, ing. Szakács Andrei — CCSIT-A Pitești.
29. **Calculatorul, programator pentru mașini unelte cu comanda numerică. Aplicații în programarea prelucrării suprafețelor paletelor rotorului hidroagregatului tip bulb KOTR 13,97—12,8** ing. V. Botez — CUG Iași.
30. **Asupra montajului asistat de calculator al dispozitivelor din elemente modulate** conf. dr. ing. A. Brăgaru, as. ing. I. Simion IPB.
31. **Utilizarea calculatorului numeric pentru acordarea optimă a regulatoarelor de temperatură la extruderele de material plastic** ing. M. Balazs, ing. G. Dimeny, ICPE Tg. Mureș.
32. **Plachete de programare pentru achiziția de date pe mini și microcalculator, folosind interfețe de proces ECAROM 881. Aplicații în industria constructoare de mașini** ing. Tothezan Puiu, ing. Tothezan Sorina, INMT.
33. **Sistem HARDWARE și SOFTWARE pentru achiziție și prelucrarea diagramei indicate de la motoare termice** ing. Tothezan Sorina — INMT.
34. **Considerații privind flexibilitatea motoarelor termice și efectul de feed-back ale datelor din exploatare în activitatea de concepție** ec. Cazimirovici Elena, ing. Tothezan Puiu, ing. Tothezan Florina, dr. ing. Săndulescu Șerban, INMT.
35. **Prelucrarea datelor obținute la încercările în tunelul trisonic INCREST** ing. Moraru Andrei, mat. Pleșa Georgeta, ing. Drăghici Alex., ing. Tudose Cezar, INCREST.
36. **Simularea unui regim stabil in funcționare a cuptorului cu arc electric (5 t oțel)** Kadar Ladislau, I. P. Cluj-Napoca.

37. **Optimizarea cutiei de viteze a unui automobil pe baza simulării pe calculator a performanţelor sale** dr. ing. Săndulescu Şerban — INMT.
38. **Modelarea matematică a curgerii turbionare izoterme cu desprindere internă, cu aplicaţie la sistem complet arzător+ambrazura — focar** ing. E. D. Cristea — ICSIT-EE, ing. M. Petcu.
39. **Program de calcul de ardere a materialului combustibil solid, funcţie de fineţea de măcinare** ing. M. Dogaru, ing. I. Müller ICSIT-EE.
40. **Metode şi tehnici de calcul şi analiza economică pe calculator utilizate în concepţia şi modernizarea motoarelor termice** ec. Cazimirovici Elena — INMT.

3. PRODUSE PROGRAM APLICATIVE CPAC ÎN ENERGETICĂ

Organizator: ICCE-ISPE, ICEMENERG

1. **Proiectarea asistată de calculator a obiectivelor energetice (centrale termo, linii, staţii). Realizări şi perspective** dr. ing. N. Chirculescu, ing. B. Ungureanu, ing. A. Marinescu — ISPE.
2. **Sistem de organizare şi gestiune automată a datelor utilizate în proiectare, la evaluarea siguranţei în funcţionare a obiectivelor energetice** ing. G. Grigoriu, ing. M. Ursu, mat. G. Stănciulescu — ISPE.
3. **Program pentru evaluarea impactului condiţiilor meteorologice asupra necesităţilor de energie electrică** dr. ing. E. Costin, ing. M. Săndulescu — ISPE.
4. **Proiectarea asistată de calculator a părţii electrice şi de automatizare a termocentralelor (sistemul CICERO)** (ing. E. Neniţă, ing. C. Apostol — ISPE.
5. **Proiectarea asistată de calculator a circuitelor secundare electrice din termocentrale (subsistemul CISEL)** ing. D. Racoviţan, mat. Gh. Toader — ISPE.
6. **Elaborarea asistată de calculator a proiectului acţionărilor dintr-o termocentrală (subsistemul MARINA).** ing. O. Apostol, mat. Gh. Toader — ISPE
7. **Realizarea pe calculator a conexiunilor din schemele desfăşurate electrice (aplicaţia LENORA).** ing. O. Apostol, mat. Gh. Toader — ISPE
8. **Determinarea configuraţiei reţelelor electrice** M. Bratu — ASG
9. **Pachet de programe pentru proiectarea asistată de calculator a reţelelor electrice industriale** ing. B. Ungureanu, ing. I. Coman — I.S.P.E., dr. ing. H. Crişciu — I.P.B.
10. **Program pentru dimensionarea instalaţiilor complexe de legare la pămînt în industriale** ing. B. Ungureanu, ing. I. Coman — ISPE, dr. ing. Crişciu — IPB. **staţiile electrice de înaltă tensiune.** ing. B. Ungureanu, ing. E. Surrim — ISPE.
11. **Stimularea unui regim stabil de funcţionare a cuptorului cu arc electric** L. Kadar — Institutul Politehnic „Traian Vuia: Timişoara.
12. **Calculul funcţiilor de transfer ale unei scheme electrice cu n noduri** ing. C. Mihail, ing. D. Cosma CCSITTc — Bucureşti
13. **Proiectarea sistemelor de termoficare asistată de calculator** ing. N. Niculescu, Ing. I. Uţă — ISPE.
14. **Oportunitatea prevederii degazării de CO_2 în instalaţiile de demineralizare a apei cu ajutorul calculatorului electric,** ing. C-tin Fodor, mat. Viorel Radu — ISPE.
15. **Calculul transformatoarelor,** mat. M. C. Andrieş, mat. E. Pascari CTCE — Botoşani.
16. **Sistem de proiectare asistată de calculator a construcţiilor speciale energetice** ing. V. Mustaţă, mat. M. Postolache — ISPE
17. **Utilizarea procedeului mixt „echivalenţa-diferenţe finite" la analiza stării de tensiune in structurile bidimensionale** prof. dr. ing. M. Diaconu, şef lucrări dr. ing. D. Gorbănescu, Institutul Politehnic — Iaşi
18. **Aspecte ale utilizării în ISPH a calculului automat în studiul structurilor pentru construcţii hidrotehnice** dr. ing. Stăncuţa Aurel, dr. ing. Ionescu Ştefan, ing. Corda Ion — ISPH.
19. **Probleme ale constituirii unui sistem de asistare informatică a cercetării energetice** ing. S. Felea — ICEMENERG.
20. **Model şi program de calcul pentru asistarea cercetării aferente analizei şi prognozei dezvoltării sistemului naţional al energiei României. Realizări, rezul-**

tate, **funcţionalitate în cercetare** dr. ing. M. Duma, mat. C. Novak, mat. Z. Frăţilă, mat. G. Stănciulescu, ec. A. Dobre ICEMENERG.

21. **Model şi program de calcul pentru corelarea prognozei dezvoltării energetice cu prognoza dezvoltării economice dr.** ing. M. Duma, ec. Gh. Pribiag, ec. A. Dobra, mat. St. Constantin, ICEMENERG.
22. **Studierea sistemelor interconectate cu ajutorul modelelor de Tip REIDIMO** Dr. ing. M. Pomârleanu, ing. L. Bejuscu, ICEMENERG.
23. **Program pentru calculul caracteristicilor instalaţiilor complexe de legare la pămînt** dr. ing. L. Goia, mat. Gh. Apetrei, ICEMENERG.
24. **Calculul şi dimensionarea circuitelor de încercare cu ajutorul programului PROPAG. RT (propagări regimuri tranzitorii).** ing. R. Enache, ICEMENERG.
25. **Program de calcul al parametrilor termodinamici ai apei şi aburul. Metode de inversare a funcţiilor termodinamice** ing. R. Regman, ing. A. Duma, ICEMENERG.
26. **Calculul parametrilor destinderii aburului în treptele unei turbine — program de calcul pentru calculatorul de buzunar programabil TI 58. Particularităţile programării pe calculator** ing. A. Duma — ICEMENERG.
27. **Model şi program de calcul pentru determinarea pe cale statistică a caracteristicilor lignitului din bazinul Rovinari şi a şistului bituminos din bazinele Anina şi Roman** ing. M. Rădulescu, ing. Gh. Chercea, chim. M. Zglobiu, ing. I. Balma — ICEMENERG.
28. **Modele şi programe de calcul pentru stabilirea proprietăţiilor fizice ale agenţilor termodinamici utilizaţi la cazanele de abur** ing. I. Balma, ing. C. Totolo — ICEMENERG.
29. **Cercetarea asistată de calculator a cîmpurilor termice** prof. dr. ing. I. Gh. Carabogdan, dr. ing. C. Brătianu, mat. M. Sterescu — IPB.
30. **Programul pentru determinarea constantelor debitmetrice la o centrală hidroelectrică pe baza măsurătorilor operative (curente) din exploatare** dr. ing. Gh. Cosma, mat. Gh. Apetrei, tehn. E. Păun — ICEMENERG.
31. **Modele matematice şi programe de calcul pentru stabilirea posibilităţilor de utilizare a energiei solare combinată cu pompe de căldură pentru încălzirea clădirilor** ing. A. Florescu, ing. E. Ursu, tehn. P. Ene, ICEMENERG, ing. C. Petrescu — ISPE, ing. D. Bercaru — IPCT.
32. **Model matematic şi program de calcul pentru stabilirea parametrilor şi eficienţei energetice şi economice la funcţionarea pompei de căldură în combinaţie cu sistemele solare pentru prepararea apei calde de consum** ing. N. Burghiu — ICEMENERG.

4. PRODUSE PROGRAM CPAC APLICATIVE DIN ALTE RAMURI INDUSTRIALE

Organizator: ICMUEE, CNST-ICI

1. **Elaborarea asistată de calculator a documentaţiei de implementare a dispozitivelor numerice** lt. col. ing. A. Drăgan, cpt. lt. ing. A. Serb, lt. ing. D. Pop — Academia Militară.
2. **Spre un sistem pentru proiectare asistată de calculator a unităţilor microprogramate** I. A. Leţia, K. Pusztai, S. Nedevschi, I. P. Cluj-Napoca.
3. **Sistem de programe pentru proiectarea asistată de calculator a structurilor de echipamente** L. Orăşanu, R. Gaşpar, F. Filip — ICI.
4. **Ciclul proiect-experiment — calcul în optimizarea opticii ionice a unui analizor magnetic de masă** C. Fătu, A. — ITIM — Cluj-Napoca.
5. **Scurtă privire asupra unor metode de amplasare optimală folosite în proiectarea asistată de calculator a echipamentelor numerice** ing. C-tin Donciu, ing. T. Lungu, ing. Al. Rădoşanu, mat. V. Domocos — CCAB.
6. **Analiza în curent continuu a circuitelor neliniare** ing. O. Mănescu — CCSITTc conf. dr. ing. D. Stanomir — IPB.
7. **Simularea pe calculator numeric a regimurilor de funcţionare normală şi de avarie pentru invertorul de divizare de timp** ing. L. Lefkovits, ICPE — Tg. Mureş.
8. **Stimularea pe calculator numeric a convertizoarelor statice de medie frecvenţa** ing. F. Jakab — ICPE — Tg. Mureş.

9. **Produs informatic pentru descrierea topologică a plăcilor cu circuite integrate** I. Parpucea, M. Popliceanu, S. Damian — IPA.
10. **Posibilități SOFTWARE și HARDWARE pentru compactarea / expandarea datelor** Gabriel Marcu — ITC, Vasile Petrovici — ICI.
11. **Model matematic și programe privind exploatarea sării prin dizolvare** ing. Tőkés — ICPMSN Cluj-Napoca, ing. Gh. Părău — ICPMSN Cluj-Napoca, conf. dr. ing. Gr. Moldovan, Fac. matematică Cluj-Napoca, progr. T. Toadere — C.C. Univ. Cluj-Napoca, Analist Gh. Mureșan — C.C. Univ. Cluj-Napoca.
12. **Estimarea secvențială a sistemelor chimice** prof. dr. ing. R. Mihail — IPB, ing I. Grozeanu — ICITPR — Ploiești.
13. **Pachet de programe pentru identificarea parametrilor cinetici în procesul de oligomerizare industrială a ciclohexenei** O. Petruș, B. Constantin, Lab. fiz. computațională — Univ. „Al. I. Cuza" Iași, E. Ionescu, Șt. Gavril, C. Niculiță D. Corduneanu, V. Rusu CCTCS Săvinești.
14. **Algoritm și program de identificare a parametrilor în modele de cinetică biochimică pentru producerea penicilinei** D. Petruș, Lab. de fizică computațională „Univ. Al. I. Cuza" Iași, V. Clocotici — C.C. univ. „Al. I. Cuza", Iași.
15. **Elaborarea pe cale instrumentală a rețelelor de vopsire** mat. M. Stan, ing. A. Stoica, teh. A. Gheorghe Inst. de cercetări textile
16. **Program pentru calculul echilibrelor complexe în chimie** Ilie Bucur Centrul de cercetări pentru îngrășăminte chimice Craiova.
17. **Realizări în cercetarea antropologică asistată de calculator** dr. C. Riscuță, I. V. Babeș, ing. G. Grigoriu, mat. A. Petre — ISPE
18. **Mapping cardiac asistat de microcalculator de proces** L. Losonczi — ICPE Tg. Mureș
19. **Automatizarea activității administrative și arhivarea documentației scrise din activitatea de proiectare** M. Sbiera, M. Ioniță — IPA.
20. **Elaborarea automată a specificațiilor și devizelor din activitatea de proiectare** D. Petrov — IPA
21. **Interpolator de tip ADN implementat cu calculator de proces** L. Losonczi — ICPE Tg. Mureș

5. PRODUSE-PROGRAM CPAC APLICATIVE DIN CONSTRUCȚII CIVILE ȘI INDUSTRIALE

Organizator: ICCPDC — IPCT

Daschevici L., Capațină D. — Bilanț al activității și orientările de perspectivă ale Comisiei ICCPDC pentru automatizarea proiectării și cercetării construcțiilor.

PROBLEME GENERALE DE INFORMATICĂ APLICATĂ

Niculiu L. — Aportul informaticii în proiectarea unor soluții eficiente în construcții.
Petrovici R. — Proiectarea asistată de calculator pentru construcții în străinătate.
Capatina D., Cornea T. — Către un sistem integrat de proiectare asistată de calculator.
Dinu V., Balosache N. — Sistem integrat de proiectare interactivă a partiurilor de arhitectură și a structurilor construcțiilor de locuințe în regim grafic și numeric.
Cristea Gh. — **„INTERGEN"** — Generator de interfețe pentru utilizarea programelor în regim interactic.
Dogaru L., Marton L. — **INFORS** — Sistem automat pentru informarea științifică și tehnică în construcții.
Pecol E. — Subprogram general pentru crearea punctelor de reluare.
Burchi M. — **GENCAR 1.**

ANALIZA ŞI PROIECTAREA STRUCTURILOR DE CONSTRUCŢII

Titaru Em., Capatina D. — Proiectarea structurilor antiseismice în cadre ductile pe baza conceptelor mecanismelor impuse de disipare a energiei, a ierarhizării formării articulaţiilor şi a solicitărilor după direcţii oblice în plan.

Catarig Al., Petrina N. — **Matricea de rigiditate şi vectorul forţelor de încastrare** perfectă la bara prinsă elastic de noduri cu dimensiuni finite.

Petrescu I., Creţu D. — Element finit plan triunghiular de placă şi membrană dezvoltat în teoria Kirchoff Discret.

Creţu D., Creţu L. T., ş.a. — Element finit de placă plană rezemată pe mediu elastic tip Winkler.

Poteraşu V. F., ş.a. — Analiza dinamică a diafragmelor prin metoda elementelor de frontieră.

Kun S., Pop S., Benke S. — Experimentări numerice pe calculator în vederea modelării structurilor cu diafragme din beton armat.

Dubina D., Eles P., Bran O. — Program de calcul pentru verificarea la stabilitate a barelor cu pereţi subţiri solicitate la compresiune cu încovoiere.

Cuteanu E., ş.a. — Analiza statică, dinamică şi de stabilitate a structurii de rezistenţă a maşinii de haldat.

Dosa A., ş.a. — Sistemul de programe „PALES".

Munteanu I., ş.a. — Răspunsul dinamic al structurilor pentru susţinerea agregatelor eoliene.

Friedrich R., ş.a. — Precizări privind utilizarea programului de calcul „PLOBA".

Kocsis M., Moţ T. — Dimensionarea grinzilor metalice cu goluri în inimă.

Jantea C., Stanciu A. — Determinarea repartiţiei solicitării la elementele unei îmbinări metalice cu ajutorul calculatorului electronic.

Serb G. A. — Optimizarea dimensionării secţiunilor dreptunghiulare din beton armat solicitate la compresiune excentrică prin teleprelucrarea în limbaj FORTRAN conversaţional.

Serb G. A., Oprea Gh. — Optimizarea dimensionării secţiunilor dreptunghiulare din beton armat solicitate la torsiune cu încovoiere.

Cismigiu Al., Dogaru L. — Analiza spectrală a seismului „Vrancea 77".

Munteanu I., Dogariu E., — Spre o concepţie unitară în mecanica structurilor.

Mustaţa V., Ungureanu A. ş.a. — Calculul structurilor complexe de mari dimensiuni cu exemplificare pe calculul static şi seismic (după P 100/81) al clădirii principale la CNE Cernavodă.

Gobesz F., Bârsan G. ş.a. — Ferme din cabluri rigidizate hobanat.

Barsan G., Alexa P. ş.a. — Răspunsul dinamic al structurilor cu reazeme depărtate supuse excitaţiilor defazate.

Friedrich R., Stoian V. ş.a. — Program pentru calculul diafragmelor asamblate din panouri mari.

Stoian V., Friedrich R., ş.a. — Asupra utilizării programului de calcul „BIOGRAF:.

Constantinescu D., Giurgea D., ş.a. — „JUMBO" — Un program de proiectare a stîlpilor de beton armat.

Sandi H., Minea S., ş.a. — Analiza parametrică asupra unei clase de mişcări seismice artificiale.

Sandi H., Borcia I. S., — Analiza parametrică asupra comportării seismice a unor structuri în cadre cu zidărie de umplutură.

Sandi H., Stancu O., ş.a. — Calculul spectrelor de acţiune de etaj. Programare şi aplicaţii.

Dogaru L., D'ALBON Gh., ş.a. — SEXT-B — Pachet de programe destinat proiectării asistată de calculator a portalelor alcătuite din elemente de beton armat centrifugat utilizate în staţiile de transformare de 110—220 KV.

Capriţa D., Jakab E. — Unele aspecte ale utilizării elementelor de margine în modelarea numerică.

Costan D., Creţu D. — Aspecte privind evoluţia programelor din familia „SAP".

Serb G. A. — Optimizarea proiectării secţiunilor dreptunghiulare din beton armat solicitate la încovoiere utilizînd teleprelucrarea în limbaj FORTRAN conversaţional.

Serb G. A. — Optimizarea dimensionării armăturii transversale la grinzi din beton armat cu secţiunea dreptunghiulară solicitate la încovoiere cu forţă tăietoare.
Constantinescu G., ş.a. — Calculul structurilor din zidărie la acţiuni seismice cu ajutorul calculatorului FELIX, FC. 1000.
Munteanu E. Gh., ş.a. — Pentru o exploatare calitativă a tehnicilor de calcul — cu exemplificare pe o teoremă de optim la compresiune excentrică oblică.
Constantinescu G., Roşca M., ş.a. — Utilizarea calculatorului FELIX FC1000 în proiectarea construcţiilor.
Florescu D., ş.a. — Studii privind alegerea unei configuraţii geometrice raţionale a braţului de 120 m la o maşină de haldat.
Bratu N.; Munteanu E. Gh. — Programe de proiectare în construcţii pentru maşina FC 128.
Dutulescu E., Cornea T. — Calculul neliniar al structurilor de beton armat.
Moraru S., Sora I. — Principiul fizic şi schema logică privind un nou model mecanic de răspuns al construcţiilor etajate la acţiuni seismice.
Chisaliţa A., Parv B. — Programul „THACS" (Time History Analysis of Cable Systems).
Cocheci T., Gobesz F. — Programe pentru calculul pierderilor de tensiune în armătura postîntinsă din grinzi continue.
Manu Gh., Dumitru Gh. — Cu privire la comportarea structurilor cu nucleu la solicitări seismice.

PROIECTAREA FUNDAŢIILOR DRUMURILOR ŞI LUCRARILOR HIDROTEHNICE.

Stănculescu I., ş.a. — Program pentru calculul pilotului izolat sub acţiunea forţelor laterale şi momentelor.
Stoica N. — Program pentru proiectarea şi desenarea fundaţiilor pahar.
Stanciu A., Boti N. — Calculul automat al masivelor din pămînt armat cu parament înclinat şi umplutura realizată din materiale coezive.
Munteanu T., Ionescu V. — Optimizarea nivelării terenurilor pentru irigarea prin scurgere la suprafaţă.
Dinescu M., ş.a. — Programe de calcul pentru determinarea volumului de terasamente şi editarea profilelor transversale ale canalelor de irigaţii.
Morlova S., ş.a. — Folosirea calculatorului electronic în procesul proiectării moderne pentru rezolvarea problemelor de demolări de canale.

CALCULE DE FIZICA CONSTRUCŢIILOR.

Calcule de fizica construcţiilor.
Moga I., Comsa E. — Calculul şi analiza cîmpului neliniar de temperatură în regim termic nestaţionar pentru elemente, cu structură complexă, cu programul CIMPNELP.
Comsa E., Moga I. — Unele consideraţii cu privire la alegerea soluţiilor constructive pentru sporirea gradului de protecţie termică a elementelor de închidere la clădiri existente.
Stoica N. — Program pentru calculul necesarului de căldură.

PROIECTAREA ASISTATĂ DE CALCULATOR: ARHITECTURA UNUI SISTEM PROTOTIP

dr. ing. G. Manolescu
I.T.C.I.

1. Introducere

Studierea procesului de proiectare în ansamblu [12, 15, 19], pune în evidenţă încadrarea sa în categoria sistemelor tehnologice şi, mai precis, în cea a sistemelor tehnologice de prelucrare a informaţiilor. De aici, se desprind următoarele:

● sistemele tehnologice, în general, sînt produse tehnice care trebuie să fie proiectate şi construite;

● procesul de proiectare, atunci cînd este implementat la nivel fizic, devine un sistem tehnologic, deci trebuie să fie proiectat şi construit.

○ Ultima observaţie este esenţială pentru noi şi a permis introducerea termenului de „inginerie a proiectării" [41].

Putem vorbi despre o „inginerie a proiectării asistate de calculator"? În acest sens, următorul citat care dă un răspuns la întrebarea pusă, ni se pare semnificativ: „inginerii sînt obişnuiţi să gîndească despre maşini. Prin analogie, un sistem PAC poate fi comparat cu o maşină. Resursele pe care le utilizează o astfel de maşină în procesul de producţie sînt, în principal, informaţiile de intrare şi informaţiile structurate sub formă de programe, iar ieşirile sînt constituite din documentaţia şi desenele proiectului de execuţie care se află într-o anumită regiune a unei baze de date. Atît un sistem PAC cît şi o maşină convenţională necesită, în timpul funcţionării, controlul sau, cel puţin, supervizarea lor de către om.

Analogia merge şi mai departe. Maşinile pot fi proiectate pentru o utilizare independentă (stand-alone). Aceasta este şi situaţia utilizării unor sisteme PAC la cheie. Dar maşinile sînt utilizate, adesea şi într-un context mai larg în care un sistem de transport stochează produsele intermediare şi le transportă de la o maşină la alta. Sistemul de transport îşi găseşte analogia, în cadrul sistemelor PAC, în sistemul de gestiune a bazelor de date. De asemenea, în cadrul sistemelor de maşini convenţionale, diversele produse intermediare sînt supuse controlului şi apoi sînt orientate în diverse poziţii înainte de a fi fixate pe maşina următoare. În cadrul sistemelor PAC, controlul intermediar poate fi realizat prin intermediul limbajelor de interogare ale bazelor de date, iar orientarea diferită a unui produs intermediar se poate obţine prin diferitele *vederi* ale aceluiaşi obiect din baza de date. Ca şi în ingineria convenţională, o maşină PAC trebuie, ea însăşi, să fie proiectată şi construită. Pentru proiectarea şi construirea maşinilor sînt necesare alte maşini (maşini unelte) sau, cel puţin, anumite unelte manuale. Şi în acest caz, în domeniul PAC se pot găsi corespondenţe. Este vorba de sisteme de programe speciale (sistemele nucleu sau sisteme de instrumente) prin intermediul cărora este facilitată proiectarea şi construirea de sisteme PAC. Un aspect particular al acestor sisteme de instrumente este marea lor adaptabilitate, în raport cu tipul de sistem PAC pe care trebuie să îl realizeze. Această caracteristică apropie sistemele de instrumente de sistemele de fabricaţie flexibilă din cadrul construcţiilor de maşini" [19].

Răspunzînd la prima întrebare, este cazul să ne întrebăm care sînt mijloacele prin intermediul cărora se poate realiza „ingineria proiectării asistate de calculator"? Este tocmai tema lucrării de faţă.

2. Sisteme nucleu

Deși ideea unor „sistem flexibile" de fabricație a produselor informatice începe să fie larg implementată abia în jurul anilor 1976 [61], ea a apărut și s-a concretizat încă din anii 1965—1967. Faptul că această idee s-a ivit și s-a aplicat mai întîi în domeniul proiectării asistate de calculator (PAC), nu este întîmplător. Intradevăr, în acest domeniu, una dintre cerințele de bază ale a plicațiilor realizate este *înalta lor interactivitate*. Plecînd de aici și pînă la aplicarea acestei interactivități chiar la realizarea unor asemenea aplicații nu a fost decît un pas, intuindu-se că, de fapt, și produsele informatice sînt tot produse tehnice a căror proiectare și construire poate fi asistată de calculator [41]. Astfel, în anul 1967, Institutul Tehnologic din Massachutts, trece la elaborarea primului sistem flexibil de realizare de produse informatice complexe din domeniul PAC, denumit ICES [56, 57]. El va fi primul reprezentant al așa numitelor „sisteme nucleu".

Reprezentanții clasei de „sisteme nucleu" au proliferat rapid. În acest mod apar: GENESYS [1], IST [34], POLO [38], DINAS [15], REGENT [34] etc. Implementarea unor asemenea sisteme, dintre care unele au ajuns la cîteva versiuni succesive (de exemplu, ICES a generat versiunile: MIT[57], PSU [51], Mc-Auto [59]) a arătat clar că trebuie făcută distincția între următoarele trei tipuri de utilizatori:

- realizatorii de sisteme PAC;
- realizatorii de aplicații PAC;
- utilizatorii finali.

Diferența dintre aceste tipuri de utilizatori va rezulta clar dacă ne vom referi, de exemplu la sistemul PAC STRUDL, realizat prin intermediul ICES [37]. Astfel, în multe domenii ale proiectării, ingineria structurală (analiza și proiectarea structurilor pe baza metodei elementului finit și a metodei elementului de frontieră) are un rol esențial. Ea apare peste tot unde apar calcule de dimensionare și verificare de rezistență pentru corpuri solide, precum și în domeniul studiului curgerii fluidelor. Dacă pînă în jurul anilor 1968, majoritatea inginerilor care aveau de rezolvat astfel de probleme, atunci cînd utilizau calculatorul își elaborau programe proprii, după această dată s-a trecut la utilizarea intensă a pachetelor de programe gen SAP, STRUDL sau NASTRAN. Pachetul de programe STRUDL a fost construit prin intermediul sistemului ICES. Realizatori de astfel de pachete de programe complexe, de uz larg, lucrează la un nivel extrem de înalt de abstractizare. Ei trebuie să posede cunoștințe aprofundate atît în domeniul de bază în care se aplică pachetul de programe (i.e. ingineria structurală) cît și în domeniul informaticii (inginerie software). În terminologia ICES, ei sînt *realizatori de sisteme PAC*. Unul dintre obiectivele de bază ale unui „sistem nucleu" este acela de a pune la dispoziția realizatorilor de sisteme PAC instrumentele informatice prin intermediul cărora acești utilizatori să-și poată realiza performant sarcinile ce le revin. Astfel de instrumente, în concepția ICES, sînt: instrumente pentru înlănțuirea flexibilă a module-program parametrizate, existente și pentru includerea în noul sistem de astfel de module-program, sisteme pentru gestiunea datelor pe termen lung și scurt, instrumente pentru generarea unor eventuale limbaje de programare orientate pe probleme (Problem Oriented Languages — POL), necesare realizatorilor de aplicații PAC atunci cînd limbajele uzuale de nivel înalt, de asemenea orientate pe probleme, cum ar fi FORTRAN-ul și PASCAL-ul pe domeniul CPAC, nu-i satisfac etc.

Realizatorii de aplicații PAC sînt reprezentați de iniginerii-informaticieni care trebuie să rezolve probleme practice. Interesul lor nu este de a da soluții generale ci acela de a găsi soluții specifice cum ar fi: analiza eforturilor care apar în timpul funcționării unui reactor nuclear. Ei vor utiliza funcțiunile generale ale unui sistem PAC sau atunci cînd acest lucru nu-i satisface, vor crea noi funcțiuni, prin intermediul limbajelor orientate pe probleme cu ajutorul cărora, în afara programării unor algoritmi proprii, vor putea să preia și module-program parametrizate din sistem. De asemenea, prin intermediul acelorași limbaje vor putea avea acces și la alte facilități ale sistemului (de exemplu, la cele aferente gestiunii datelor).

Din punctul de vedere al *utilizatorilor finali* (marea masă a cercetătorilor și proiectanților), interesul acestora este de a folosi, într-o manieră cît mai comodă, un

sistem PAC, precum și aplicațiile dezvoltate din acesta. Pe vremea cînd a fost conceput sistemul ICES, problemele legate de „accesibilitate" („user friendly" în literatura anglo-saxonă) abia erau întrevăzute. Anii '80 au marcat o adevărată explozie în acest domeniu. Prin urmare, în conformitate cu această nouă tendință, printre instrumentele pe care trebuie să le posede un „sistem nucleu" trebuie să introducem și generatoarele de interfețe de „dialog om-calculator" prin intermediul cărora, realizatorii de sisteme și aplicații PAC elaborează astfel de componente într-o manieră cît mai apropiată de modul uzual de gîndire și exprimare a utilizatorului neinformatician. În spatele unor astfel de generatoare stau însă alte instrumente, de tip limbaje de interacțiune și dialog „om-calculator" care constituie realizările cele mai recente din domeniul limbajelor orientate pe probleme [58, 71].

O altă tendință, recentă [15], din domeniul PAC, care continuă, extinde și îmbunătățește ideia de „sistem nucleu", este introducerea noțiunii de „sistem prototip".

O remarcă importantă: dacă conceptul de „sistem nucleu" poate și, de fapt, a și fost aplicat și în afara PAC (vezi, de exemplu [48]) în schimb conceptul de „sistem prototip", în sensul în care el a fost definit în lucrarea [15] și pe care vom căuta să-l pune în evidență în continuare, este un concept specific PAC.

Conceptul de „sistem prototip" se bazează pe cîteva caracteristici valabile pentru orice sistem PAC orientat spre obținerea proiectului de execuție a unui nou produs tehnic. Și aceasta, independent de complexitatea produsului: o cutie de viteze, o uzină sau un combinat industrial. Aceste caracteristici sînt:

1. Imaginea noului produs se constituie dinamic, în cursul procesului de proiectare, de la concept și proiect de ansamblu, pînă la imaginea de detaliu și proiect de execuție. Rezultatele etapelor prin care trece, succesiv, această imagine, trebuie să fie stocate într-o așa numită „bază de date a proiectului". O asemenea bază de date, prezintă o deosebire esențială față de bazele de date cu care eram obișnuiți în cadrul gestiunii economice. Astfel, dacă structura datelor, în cadrul unei baze de date din gestiunea economică, odată stabilită, rămînea „înghețată" pe toată durata de viață a acesteia, în schimb, în cadrul unei „baze de date a unui proiect", lucrurile se petrec cu totul invers. Și aceasta deoarece, imaginea noului produs este în continuă schimbare pînă să ajungă la forma sa finală, ceea ce implică și o restructurare dinamică a datelor din „baza de date a proiectului".

2. Pentru realizarea proiectului unui nou produs tehnic, este necesar să se dispună de o serie de cunoștințe specifice, cum ar fi: materialele (cu caracteristicile lor asociate) din care vor fi realizate componentele produsului, standardele în vigoare legate de realizarea produsului, informații de natură tehnologică, informații asupra unor proiecte similare deja realizate etc. Toate aceste informații pot fi incluse într-o „bază de date tehnice". O asemenea bază de date are caracteristici similare cu băncile de date uzuale și nu ridică probleme noi.

3. Într-un anumit sens, programele din cadrul unui sistem PAC, ca, de altfel, și programele oricărui sistem informatic, pot fi considerate, din punctul de vedere al calculatorului, tot de natura unor „date". Însăși existența și rolul componentelor de tip „bibliotecar" ale sistemelor de operare confirmă acest lucru. Prin urmare, aceste programe pot fi stocate într-o așa numită „bază de programe" care poate avea rolul unei biblioteci de programe cu funcții extinse.

4. În fine, datorită punternicului caracter interactiv presupus de un sistem PAC, apare necesitatea realizării unui monitor de proces, interacțiune și dialog. Deci unui astfel de monitor îi revine sarcina de integrare a sistemului.

Din enumerarea caracteristicilor de bază ale conceptului de „sistem prototip PAC" se poate observa că ultimile două nu sînt totuși specifice sistemelor PAC. Ceea ce conferă totuși specificitatea conceptului de care ne ocupăm este *ansamblul* caracteristicilor sale.

În ceea ce privește corelarea conceptelor de „sistem nucleu" și „sistem prototip PAC", vom putea observa, în paragrafele următoare, cum ele se combină într-un nou concept care le înglobează și anume, cel de „sistem prototip PAC generalizat", pe scurt SPG-PAC, concept pe care l-am introdus în lucrarea [40].

La noi în țară, implementarea unui sistem prototip PAC generalizat a început în anul 1982.

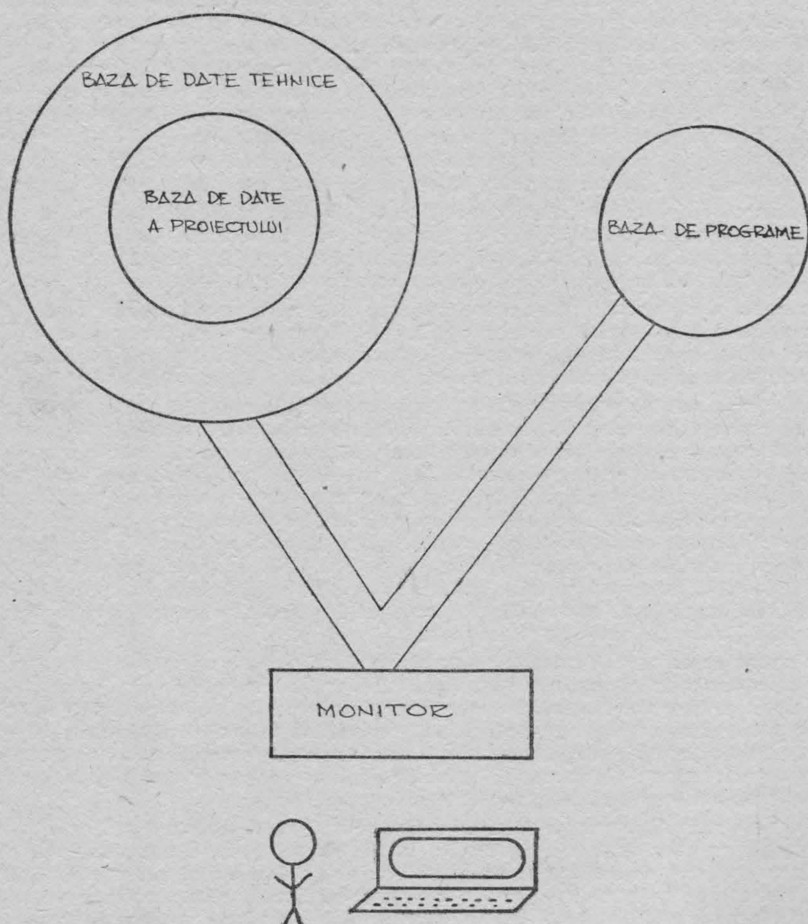


FIGURA 1 COMPONENTELE UNUI SISTEM PROTOTIP PAC

3. Ideia de prototip

Ideea de prototip este strîns legată de cea de flexibilitate, chiar și la nivel intuitiv. Ea capătă, în practica curentă, o multitudine de forme cu caracter mai concret sau mai abstract.

Astfel, în sfera realizării produselor tehnice, mai ales pentru unele produse complexe, înainte de a se trece la fabricarea industrială a acestora, se realizează, pe baza proiectului de execuție, un *prototip* al produsului. El reprezintă un prim exemplar al produsului și servește la realizarea tuturor testelor în regim normal sau de excepție de funcționare pentru punerea în evidență a funcțiunilor și performanțelor produsului. Aceste teste au menirea de a scoate în evidență modificările ce mai trebuie să fie făcute înainte de intrarea produsului în fabricație curentă. Prin urmare, chiar sub această formă, ideea de prototip este asociată cu o anume *flexibilitate* dar și cu o anumită *invarianță*. Mai departe, datorită caracterului de simulare pe care îl prezintă însuși procesul de proiectare orientat spre obținerea de noi produse tehnice (acest proces simulează o acțiune — cea de realizarea a unui nou produs tehnic — înainte ca aceasta să aibă loc), procesul implică construirea de machete, modele analogice sau modele numerice de simulare și, evident, realizarea de încercări pe asemenea entități. Astfel de machete și modele au început să intre, de asemenea, sub incidența noțiunii de prototip. Prin urmare, deși se păstrează cele două caracteristici complementare: flexibilitatea și invarianța, noțiunea de prototip capătă o formă mai abstractă. Pentru a vedea cum sînt realizate cele două caracteristici (ce par contradictorii) în cazul unui model analogic, vom da un exemplu. El se referă la studierea curgerii fluidelor prin sisteme de conducte cu ajutorul unor circuite electrice. Se știe că, atît curgerea fluidelor cît și transmisia curentului electric sînt descrise de aceleași ecuații matematice (partea invariantă); în schimb, modul de concretizare practică a fenomenelor descrise de astfel de ecuații: curgerea fluidelor prin sisteme de conducte sau transmisia curentului electric prin circuite, poate diferi (partea flexibilă).

O implementare și mai abstractă a noțiunii de prototip o găsim în cadrul inteligenței artificale [76]. Astfel, plecîndu-se de la cele două caracteristici: invarianță și flexibilitate, s-a generalizat și, în același timp, s-a abstractizat și mai mult noțiunea de care ne ocupăm, prin asocierea ei cu noțiunea de funcție. Astfel, partea constantă (invariantă) a unei funcții este considerată *forma* acesteia, iar partea variabilă (flexibilă) *mulțimea valorilor* pe care le poate lua funcția prin înlocuirea argumentelor cu valori din domeniul de definiție. De aici a derivat și s-a înrădăcinat noțiunea de *instanțiere* a prototipului. Ea corespunde unei valori particulare pe care o ia funcția pentru o valoare fixată a argumentului/argumentelor. Generalizînd și mai mult, se poate spune că instanțierea unui prototip reprezintă forma specifică pe care o poate căpăta o asemenea entitate atunci cînd partea sa variabilă a fost particularizată.

Reținem din cele expuse pînă acum, cele două caracteristici complementare ale unui prototip: *invarianța* și *flexibilitatea* sa. Aceste două caracteristici conduc la ideea că un sistem prototip generalizat PAC ar trebui să dispună de o parte invariantă, independentă de domeniul pe care operează un astfel de sistem și o parte flexibilă care să permită instanțierea prototipului pe diverse domenii particulare ale PAC.

Revenind la principalele caracteristici ale unui sistem PAC, scoase în evidență în lucrarea [15] și prezentate în paragraful anterior, acestea ne ajută să schițăm, sub formă grafică, așa cum se prezintă în figura 1, partea *invariantă* a unui sistem prototip generalizat PAC. De ce *generalizat?* (Introducerea conceptului de „sistem prototip generalizat PAC" ne aparține [39, 40]). Pentru aceasta trebuie să expunem cîteva considerații suplimentare.

În primul rînd, un sistem PAC, specific unui anumit domeniu, ca, de altfel, orice produs informatic, este tot un *produs tehnic*. Deci el va putea și va trebui să fie proiectat și construit ca orice produs tehnic [41]. Acest lucru se poate realiza și cu ajutorul calculatorului. Prin urmare, putem vorbi de o proiectare asistată de calculator a sistemelor de proiectare asistată de calcluator. Este și obiectivul pe care îl

urmăresc „sistemele nucleu“ prin intermediul instrumentelor informaitce pe care le pun la dispoziţie.

În al doilea rînd şi ca o consecinţă a primei concluzii, înseamnă că atît „sistemul nucleu“, care generează un sistem particular PAC, cît şi sistemul particular PAC generat, trebuie să fie caracterizate de acelaşi *invariant*. Un astfel de invariant îl constituie cele patru caracteristici de bază ale unui sistem prototip PAC, puse în evidenţă în lucrarea [15] şi care vor trebui să ne regăsească atît în cadrul unui „sistem nucleu“ cît şi în cadrul sistemelor PAC particulare generate de aceasta.

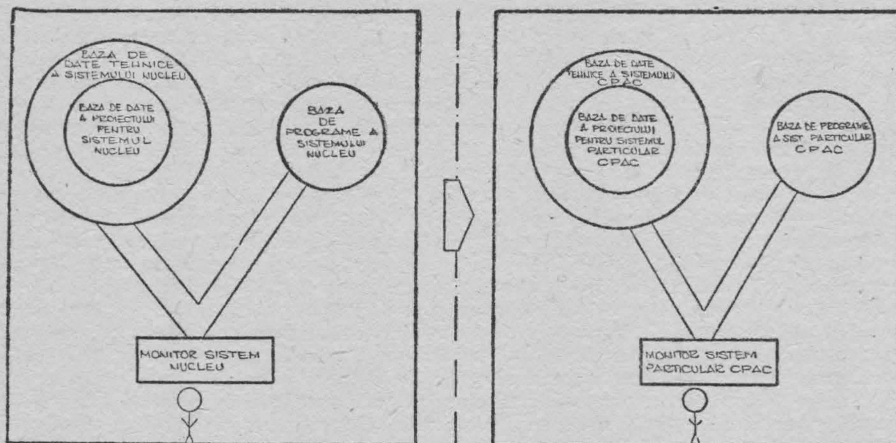


FIGURA 2. INVARIANTA SINTACTICĂ A UNUI SISTEM PROTOTIP GENERALIZAT PAC

În al treilea rînd, din concluzia anterioară derivă imediat faptul că trebuie să existe o simetrie perfectă între *forma* sistemului nucleic şi *forma* sistemelor PAC particulare generate de acesta, lucru care este pus în evidenţă în figura 2.

În fine, ceea ce diferă la un „sistem nucleu“ în comparaţie cu sistemele particulare PAC generate, este conţinutul şi semantica prezentată de elementele constitutive ale celor două tipuri de sisteme. Prin analogie, această semantică poate fi considerată similară cu valorile pe care le pot lua argumentele unei funcţii. Aceste semantici diferite, a unui „sistem nucleu“ şi, respectiv, a unui sistem particular PAC generat de acesta (de exemplu un sistem PAC pentru proiectarea de angrenaje), sînt puse în evidenţă în tabelul 1.

Ca o concluzie generală a celor discutate, se poate spune că, în cadrul conceptului de „sistem prototip generalizat PAC“, *sintaxa* „sistemului nucleu“ este *identică* cu *sintaxa* sistemelor PAC particulare generate de acesta, iar *semantica* „sistemului nucleu“ diferă de semanticile sistemelor PAC particulare generate, semantici care se deosebesc şi ele una de alta.

Tabelul 1

Nr. crt.	Denumire componentă	Conţinut componentă	
		sistem nucleu	sistem particular PAC pentru proiectare de angrenaje
1.	BAZA DE DATE TEHNICE	● date asupra unor componente standard (module-program reutilizabile)	● date asupra unor componente standard de angrenaje (roţi dinţate, arbori etc.)

Tabelul 1 (cntinuare

		• date asupra unor sisteme particulare PAC realizate • date tehologice de contruire a sistemelor particulare PAC	• date asupra unor proiecte de angrenaje deja realizate • date tehnologice de construire a angrenajelor
2.	BAZA DE DATE A PROIECTULUI	• structura și dinamica sistemului particular PAC pentru proiectare de angrenaje	• structura și dinamica unui angrenaj care se proiectează prin intermediul sistemului PAC particular
3.	BAZA DE PROGRAME	• Instrumente ale mediilor de programare • Sistem de descriere, stocare și modificare a desenelor • Sistem de gestiune a bazelor de date • Sistem de inlănțuiri flexibile și extensibile • Generator de monitoare și interfețe de dialog • Module-program reutilizabile (biblioteci tehnico-științifice) • Generator de limbaje orientate pe probleme • Limbaj/limbaje de construire a sistemelor PAC particulare	• Programe utilizator pentru proiectare de angrenaje • Sistem de grafică orientat spre proiectarea de angrenaje • Sistem de gestiune a bazelor de date • Limbaj/limbaje orientate pe probleme
4.	MONITOR	• Monitor al sistemului nucleu	• monitor al sistemul PAC de proiectare angrenaje.

4. Principalele componente și stadiul realizării unui sistem prototip generalizat PAC

Pe baza datelor din tabelul 1 și a figurii 2, în figura 3 se prezintă schița de arhitectură a unui sistem prototip generalizat PAC.

Schița de arhitectură prezentată în figura 3, este autoexplicativă așa că nu vom mai insista asupra ei.

În realizarea proiectului SPG-PAC, s-a urmărit elaborarea instrumentelor care formează sistemul nucleu și integrarea acestora, avînd ca suport hardware — minicalculatoarele.

Avînd în vedere faptul că, sistemul nucleu este un sistem de mari dimensiuni, s-a adoptat strategia realizării sale pe componente, în mod eșalonat. La baza acestei strategii au stat următoarele principii:

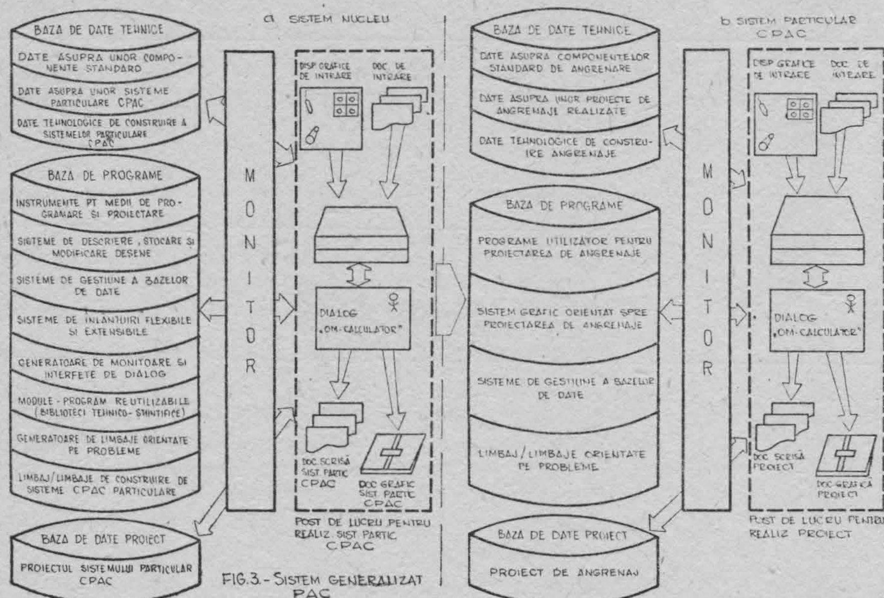


FIG. 3. - SISTEM GENERALIZAT PAC

- componentele, realizate independent, să fie ușor integrabile în sistem, dar să poată fi utilizate și ca produse-program de sine stătătoare;
- realizarea componentelor să respecte prioritățile dictate de următoarele criterii:
  — prioritățile stabilite pe baza solicitărilor beneficiarilor;
  — stadiul și rezultatele cercetărilor la noi în țară și pe plan mondial;
  — resursele: financiare și de forță de muncă, disponibile.

Prin luarea în considerare a criteriilor enunțate, s-a ajuns la următoarea eșalonare:

Prioritatea I-a:
- sisteme de descriere, stocare și modificare a desenelor (sisteme de grafică);
- sisteme de înlănțuiri flexibile și extensibile;
- generatoare de interfețe de dialog „om-calculator";
- instrumente pentru medii de proiectare;
- module-program reutilizabile.

Prioritatea a II-a:
- instrumente pentru medii de programare.

Prioritatea a III-a:
- sisteme de gestiune a bazelor de date specifice PAC;
- generatoare de limbaje orientate pe probleme;
- generatoare de monitoare.

4.1. Instrumente din prima grupă de prioritate

4.1.1. Sisteme grafice

Așa cum rezultă din literatură [22, 28], există două tipuri de aplicații grafice:
— aplicații cu ieșiri grafice pasive;
— aplicații de grafică interactivă.

Aplicațiile cu ieșiri grafice pasive se caracterizează prin faptul că desenul se realizează fără ajutorul utilizatorului final (operatorului).

Aplicațiile de grafică interactivă se caracterizează prin faptul că, rezultatul ieșirilor grafice este controlat (și modificat), în timp real, de către operatorul unei stații grafice.

Gradul de interactivitate al unei aplicații poate varia de la facilități elementare, gen „selectare de opțiuni" pe bază de meniuri, pînă la facilități complexe și complete de grafică care conțin, pe lîngă funcția de selectare și funcțiile: introducere directă de coordonate de puncte, schițări directe, plasări sau ștergeri de porțiuni ale desenelor într-o manieră apropiată operatorului uman, toate acestea fiind realizate prin intermediul display-urilor grafice și a dispozitivelor grafice de intrare (cursor cruce, joystick, tabletă grafică etc.).

Realizarea de sisteme de grafică, pe baza cărora să se poată construi rapid aplicații particulare ,a fost orientată, în cadrul SPG-PAC, pe *grafică interactivă.*

Un alt criteriu de clasificare a produselor-program de grafică, este modalitatea informatică prin care se execută funcțiile grafice. Astfel, se pot distinge:

— funcții grafice realizate prin limbaje procedurale de nivel înalt;

— funcții grafice realizate prin descrieri semi-procedurale, utilizînd interpretere.

În cadrul primei categorii, funcțiile grafice se realizează printr-un set de subrutine, stocate într-o bibliotecă, apelabile prin intermediul limbajelor procedurale de nivel înalt.

Principalul dezavantaj al acestui mod de realizare a funcțiilor grafice, constă în dificultatea elaborării programelor de aplicații și, îndeosebi, în maiera greoaie în care un desen, odată realizat, poate fi, ulterior, modificat. Într-adevăr, utilizarea unui limbaj de nivel înalt pentru realizarea funcțiilor grafice, presupune:

— scrierea, compilarea, linkeditarea și lansarea în execuție a cîte unui program pentru fiecare desen în parte;

— modificarea, recompilarea relinkeditarea și lansarea din nou în execuție a programului ori de cîte ori apare necesitatea realizării unor modificări, chiar minore, ale desenului inițial.

Un alt dezavantaj al acestui mod de realizare a funcțiilor grafice, constă în faptul că el nu prezintă facilități explicite de segmentare a desenelor (lucrul cu subdesene). Segmentarea poate fi realizată, în acest caz, prin scrierea unor subrutine asociate fiecărui segment de desen și prin asamblarea acestora în cadrul programe. Această manieră de lucru face extrem de dificilă, dacă nu imposibilă, modificarea (actualizarea) segmentelor (subdesenelor).

Dezavantajele enumerate sînt complet înlăturate în cadrul celei de a doua modalități de realizare a funcțiilor grafice. Componentele principale ale unui produs-program realizat în conformitate cu această modalitate, sînt:

— o bibliotecă de module care realizează funcțiile grafice primitive (trasare de linii, cercuri, arcuri de cerc, hașurări, prisme, conuri, trunchiuri de con și de prismă, sfere, înlăturarea liniilor ascunse, proiecții etc.);

— un pogram tip interpreter;

— un fișier de descriere a desenelor.

Fișierul de descriere a desenelor conține, în ordinea secvențială a realizării unui desen, codul mnemonicelor modulelor din biblioteca de primitive, precum și parametrii ceruți pentru activarea acestor module.

Programul interpreter citește fișierul de descriere, verifică corectitudinea fiecărei descrieri și lansează în lucru modulele de realizare a funcțiilor grafice.

Pentru realizarea segmentării desenelor, în cadrul unui fișier de descriere, se poate introduce un cod de mnemonică specială (de exemplu SEGMENT). La întîlnirea acestui cod, programul interpreter face apel la un subfișier de descriere în care această mnemonică este, la rîndul său, expandată în mnemonice ale modulelor cu funcții primitive de grafică sau mnemonice de tip SEGMENT. În acest mod este posibilă realizarea unei segmentări, teoretic nelimitată, a unui desen.

De reemarcat faptul că acest mod de realizare a funcțiunilor grafice nu exclude posibilitatea apelării modulelor (subrutinelor) de primitive în cadrul unor programe de nivel înalt. Prin urmare, biblioteca de primitive poate fi utilizată și în acest mod de realizare a funcțiilor grafice.

În cadrul modului de realizare a funcțiilor grafice prin descrieri semi-procedurale, apare evidentă ușurința realizării unui desen chiar de către utilizatori cu cunoștiințe elementare de programare. În plus, atît la crearea desenului inițial, cît și în momentele modificării acestuia, se evită complet scrierea (rescrierea), compilarea (recompilarea) și linkeditarea (relinkeditarea) programelor.

Realizarea de sisteme de grafică, pe baza cărora să se poată construi rapid aplicații particulare, a fost orientată, în cadrul SPG-PAC, spre realizarea funcțiilor grafice prin *descrieri semiprocedurale*.

În momentul de față, în cadrul proiectului SPG-PAC, a fost realizat și este operațional, un sistem de descriere, stocare și modificare a desenelor în plan și spațiu. El este compus din: INTERPLOT — pentru realizarea desenelor pe mese de desen și TRIDENT — pentru realizarea și punerea la punct a desenelor pe display-ul grafic DAF 2020. Sistemul a fost realizat conform standardului ANSI SIGGRAPH'79. În afara altor dezvoltări ale sistemului (de exemplu adaptarea sa pentru monitoare color), se are în vedere și reproiectarea sa conform standardului ISO, GKS'82 [22] și PHIGS [80], standard ANSI'84.

4.1.2. Sisteme de înlănțuiri flexibile și extensibile

Realizarea unor asemenea sisteme, care vizează crearea posibilității de alcătuire dinamică de lanțuri din module-program reutilizabile existente într-o bibliotecă, precum și a posibilității de adăugare de noi module-program în bibliotecă poate urma două căi:

— utilizarea unui limbaj procedural extensibil de nivel înalt;

— utilizarea unor interfețe de interacțiune și dialog „om-calculator".

Ambele soluții pot realiza funcțiile avute în vedere și, în măsura în care sînt implementate într-o manieră „user-friendly", prezintă aceleași avantaje.

În cadrul proiectului SPG-PAC, s-au abordat ambele căi.

Pe direcția limbajelor procedurale extensibile de nivel înalt, a fost realizat limbajul EDL, de tip FORTRAN care însă, prin extensibilitatea sa, în sensul posibilității adăugării unor noi enunțuri și prin accesibilitate (similară cu a limbajului BASIC — în plus, enunțurile EDL sînt în limba română), s-a dovedit extrem deadecvat scopurilor pentru care a fost creat.

Pe cea de a doua direcție, cea a utilizării unor interfețe de interacțiune și dialog, a fost realizat produsul-program SINTRAC. El prezintă numai funcția de înlănțuire de module-program reutilizabile. A doua versiune a produsului, care se află în lucru, va realiza și funcția de adăugare de noi module-program precum și facilități suplimentare pentru utilizator: posibilitatea interogării unei baze de date în care sînt descrise funcțiile modulelor-program existente, parametrii acestora etc., auto-instruire în utilizarea produsului, facilități tip „help" sporite etc., toate acestea realizate în manieră „user-friendly" [62].

4.1.3. Generatoare de interfețe de dialog „om-calculator"

În cadrul interacțiunii dintre om și calculator, ambii parteneri trebuie să facă eforturi pentru a asigura o cooperare strînsă. Bariera care apare, se datorează faptului că, în cadrul unei asemenea cooperări, calculatorul poate da răspuns la o diversitate mare de întrebări puse de om, dar asemenea întrebări trebuie să fie formulate într-o manieră foarte precisă, utilizînd moduri canonice de exprimare care sînt destul de îndepărtate de modul uzual de gîndire și exprimare al utilizatorului obișnuit (utilizator final). Prin urmare, singura soluție posibilă este ca, odată cu punerea la dispoziție a unui sistem PAC particular sau a aplicațiilor dezvoltate pe baza acestuia, să se pună la dispoziția utilizatorilor (finali) și facilități speciale care să asigure cooperarea menționată. Astfel de facilități sînt materializate în așa numitele „interfețe de dialog om-calculator".

„Sistemele interactive înregistrează o popularitate din ce în ce mai accentuată; în același timp însă, tehnicile și instrumentele pentru definirea diverselor aspecte ale interactivității, au rămas în urmă" [64]. Acest lucru este îndeosebi adevărat pentru dialogul om-calculator. Mai trebuie adăugat că, înainte de a se pune problema

realizării unui generator automat de interfețe om-calculator, generator ce reprezintă un „instrument“ (informatic), este necesar să se găsească o tehnică adecvată și bine formalizată de definire a unor asemenea interfețe. Pe baza unei asemenea tehnici este posibil să se realizeze descrieri care să reprezinte „intrări“ pentru un astfel de generator. Acesta este și motivul pentru care, în cadrul proiectului SPG-PAC, eforturile au fost orientate, în prima etapă, spre constituirea unei asemenea tehnici.

În acest sens, s-a plecat de la următoarele constatări:

- limbajele de programare de nivel înalt (FORTRAN, PASCAL etc.) au fost proiectate, în principal, pentru a asigura un „monolog“ (ele nu posedă facilități explicite pentru implementarea „dialogului“). Cu alte cuvinte, „omul“ comandă și „calculatorul“ execută. În acest mod, evident, dialogul nu poate să apară.
- Anumite tehnici ce au fost propuse pentru definirea dialogului „om-calculator“, nu sînt încă utilizate în mod curent. În același timp, se poate observa existența unei mari diversități a acestor tehnici, lucru care sugerează o rezolvare încă nesatisfăcătoare a problemei.

Plecînd de la dezavantajele pe care le prezintă tehnicile existente, Shneiderman [64] a propus utilizarea așa numitelor „gramatici multipartidă“ pentru a defini atît interacțiunea cît și dialogul „om-calculator“.

Deși lucrarea lui Shneiderman este extrem de interesantă, ea prezintă numai o schiță a gramaticilor multipartidă și nu o completă formalizare a acestora.

Avînd la bază lucrarea lui Shneiderman, în cadrul proiectului SPG-PAC, s-a pus la punct o tehnică riguroasă de definite a interfețelor „om-calculator“. Deoarece ea a fost prezentată, într-o primă variantă, în lucrarea [42], nu o vom mai expune. Trebuie să menționăm că această tehnisă a fost, pînă în prezent, testată în cadrul realizării a 22 de produse informatice, dintre care 14 din domeniul PAC. Testarea s-a făcut atît de către autori, cît și de alte echipe de realizare de produse informatice. Rezultatele, mai mult decît încurajatoare, au permis trecerea la elaborarea unui generaor de interfețe de „om-calculator“ bazat pe acestă tehnică de definire. Acest generator este în curs de realizare.

4.1.4. Instrumente pentru medii de proiectare

Un *mediu de proiectare* al unui sistem particular PAC, este constituit din resursele de tip uman, software și hardware implicate în procesul de proiectare, începînd de la formarea „conceptului“ și terminînd cu proiectul fizic al sistemului (din „proiectare“ nu face parte etapa de „construire“ a sistemului, reprezentată de elaborarea și testarea programelor).

Sistemele PAC au rolul de a introduce asistența calculatorului în proiectarea produselor tehnice. La rîndul său, un sistem PAC este, așa cum s-a arătat, el însuși, un produs tehnic a cărui proiectare poate fi asistată de calculator. Noțiunea de mediu de proiectare evidențiază tocmai acest lucru. Din punctul de vedere al părții „software“, instrumentele informatice sînt limbajele de descriere a sistemelor (informatice) și analaizoarele asociate. Aceste instrumente, lansate în deceniul trecut în cadrul proiectului ISDOS [69], sînt bine cunoscute și la noi în țară prin realizarea produsului LDS/ADES (cu ultima variantă — PACSIN) pentru calculatoarele de capacitate medie FELIX, așa că nu vom mai insista asupra lor. Trebuie totuși să menționăm că ele sînt indispensabile pentru descrierea unui sistem particular PAC în baza de date a proiectului ce aparține sistemului nucleu.

În prezent, în cadrul proiectului SPG-PAC este în curs de realizare varianta pe minicalculator a unor asemenea instrumente. Această variantă, față de PACSIN, prezintă îmbunătățiri substanțiale, inclusiv interfețe „om-calculator“.

4.1.5. Module-program reutilizabile.

Astfel de module-program sînt cele de natura unor programe matematice și tehnico-științifice de uz general acoperind domenii ca: rezolvarea de ecuații (sisteme liniare, diferențiale ordinare, diferențiale cu derivate parțiale, calcul cu matrici etc.), statistică, modelare-simulare, prelucrare date experimentale etc.

În cadrul proiectului SPG-PAC au fost realizate, îmbogățind bibliotecile existente, un număr de aproximativ 800 de noi componente. Realizarea acestor noi componente a avut drept modele de referință componente ale unor biblioteci de programe bine cunoscute pe plan mondial, cum ar fi IMSL, FUNDPAC etc. În continuare, se elaborează noi astfel de module-program.

4.2. Instrumente din a doua grupă de prioritate

4.2.1. Instrumente pentru medii de programare

Un *mediu de programare* (construire) a unui sistem particular PAC, în sensul larg al noțiunii de „mediu de programare", este constituit din resursele de tip uman, software și hardware necesare construirii sistemului.

Dacă luăm în considerare acest sens larg al definiției unui mediu de programare a unui sistem particular PAC, înseamnă că în resursele „software" trebuie să includem tot setul de instrumente ce alcătuiesc sistemul nucleu, cu excepția limbajelor de descriere a sistemelor și a analizatoarelor asociate.

În literatură însă, prin „mediu de programare" (în sens restrîns, adăugăm noi) se desemnează *numai* instrumentele aferente realizării programelor într-unul dintre limbajele, de obicei de nivel înalt, de programare: COBOL, FORTRAN, PASCAL etc. Astfel de instrumente sînt: instrumente pentru verificarea corectitudinii programelor, depanatoare de erori, editoare de texte pentru scrierea și modificarea programelor în format sursă etc. [17, 47, 65]. Acest sens al noțiunii de „mediu de programare" îl vom utiliza în continuare.

Limbajele de programare de nivel înalt adecvate PAC, sînt FORTRAN și PASCAL.

Se poate considera că, într-o primă etapă, trebuie acordată prioritate limbajului FORTRAN datorită utilizării sale largi. Mai mult, unele prognoze [79] consideră că limbajul FORTRAN va continua, cel puțin pînă în anul 2000 să fie folosit extrem de intens de marea masă de programatori. PASCAL-ul, ale cărui avantaje, în raport cu FORTRAN-ul și alte limbaje de programare se cunosc, este candidatul imediat următor, aceleași prognoze considerînd că el, alături de ADA, vor deveni limbajele uzuale ale viitorului pentru informatica convențională (deci, făcînd abtracție de inteligența artificială).

Acestea au fost motivele pentru care, din acest an, în cadrul proiectului SPG-PAC, au fost începute cercetările în domeniul „mediilor de programare FORTRAN". În etapa următoare, vor fi vizate „mediile de programare PASCAL". În cadrul mediilor de programare FORTRAN, particularizate pe domeniul PAC, cercetările nu sînt prea avansate nici pe plan mondial [79]. Cu atît mai puțin în domeniul mediilor de programare PASCAL, specifice PAC.

4.3. Instrumente din a treia grupă de prioritate

Stadiul cercetărilor, pe plan mondial, în legătură cu instrumentele din a treia grupă de prioritate: sisteme de gestiune a bazelor de date specifice PAC, generatoare de limbaje orientate pe probleme și generatoare de monitoare pentru sisteme particulare PAC, se află la nivelul inventarierii problemelor care trebuie să fie rezolvate. Aceste probleme par destul de spinoase, așa cum se va vedea în continuare.

4.3.1. Sisteme de gestiune a bazelor de date specifice PAC

Așa cum rezultă din literatura de specialitate [2, 5, 7, 24, 35, 46, 50, 54, 73], principalele probleme privind organizarea și reprezentarea datelor în domeniul PAC, sînt următoarele:

● asigurarea unei flexibilităţi ridicate a structurilor de date, atît în ceea ce priveşte posibilităţile de reprezentare a tuturor tipurilor de relaţii (unu la unu, unu la mai mulţi, mai mulţi la unu şi mai mulţi la mai mulţi) cît şi în ceea ce priveşte schimbarea, cu frecvenţă mare în timp, a unor structuri de date;

● asigurarea posibilităţii de a se începe procesarea datelor din orice nod al structurii;

● necesitatea introducerii unei ierarhizări în structurarea datelor;

● asigurarea posibilităţii de a se forma rapid subscheme potrivit unor puncte de vedere particulare, puncte de vedere ce nu pot fi, de cele mai multe ori, prevăzute în timpul proiectării unei baze de date;

● asigurarea posibilităţii de parcurgere a unei structuri ierarhizate de date de „sus îs jos" (top-down), de „jos în sus" (bottom-up) şi în mod „mixt";

● găsirea unui mod unitar de reprezentare a datelor de naturi diferite: numerice, alfanumerice şi grafice.

Toate aceste cerinţe ridică probleme serioase în faţa sistemelor de gestiune a bazelor de date existente, construite, mai ales, pentru realizarea unor aplicaţii de gestiune economică sau pentru prelucrare de texte şi informare-documentare, domenii în care multe dintre cerinţele enumerate nu apar.

Interesant este de consemnat, în încheiere şi o concluzie referitoare la utilizarea diverselor tipuri de structuri existente (arborescenţă, reţea şi relaţională) în domeniul PAC. Astfel, în [7] se trage concluzia că utilizarea acestor tipuri de structuri necesită, pentru satisfacerea unor cerinţe de genul celor consemnate, elaborarea unor algoritmi cu complexitate extrem de ridicată, ceea ce conduce la mărirea substanţială a timpului de răspuns, deci la o performanţă scăzută din acest punct de vedere, lucru, de multe ori, greu de acceptat în cazul unor sistem PAC ce solicită o înaltă interactivitate şi răspunsuri, de la calculator, „în timp real".

4.3.2. Generatoare de limbale orientate pe probleme.

Limbajele orientate pe probleme, tind să devină limbaje de „dialog", în opoziţie cu limbajele tip „monolog" de nivel înalt actuale: PASCAL, ADA, FORTRAN, COBOL etc. şi de nivel foarte înalt (proprii inteligenţei artificiale) LISP, PROLOG, KRL etc. [53, 64]. Referinţele din literatură în acest domeniu sînt destul de sărace şi ele marchează abia punerea problemei.

2.3.3. Generatoare de monitoare de dialog pentru sisteme particulare CPAC.

Înainte de a se pune problema realizării unor asemenea generatoare, trebuie găsite soluţiile corespunzătoare satisfacerii cerinţelor specifice ridicate în faţa monitoarelor de sisteme particulare PAC.

Dacă vom considera că rolul unui monitor de sistem particular PAC este de a asigura transformarea funcţiilor externe ale sistemului (aşa cum sînt văzute acestea de utilizatorul final) în funcţii interne (realizate de calculator) şi invers, precum şi de a armoniza funcţionarea internă a sistemului, atunci tot în sarcina monitorului intră şi supervizarea interfeţelor „om-calculator". În această situaţie, cerinţele pe care trebuie să le satisfacă un monitor al unui sistem particular PAC de complexitate ridicată, sînt [4, 19, 23, 29, 33]:

● realizarea unui timp de răspuns acceptabil (lucrul în „timp real") în mod de exploatare interactiv, prin preluarea de către monitorul sistemului a unor funcţii îndeplinite, de obicei, de către sistemele de operare;

● asigurarea posibilităţii ca sistemul să poată răspunde la întrebări nestandard ale utilizatorilor finali;

● implementarea unor mecanisme de „back-traking", de structurare dinamică a informaţiilor şi de introducere interactivă de date de intrare uzuale şi de criterii de decizie pentru alegerea de variante de soluţii în cursul procesului de proiectare;

● asigurarea posibilităţii de includere uşoară a unor noi module (programe) în sistem şi a posibilităţii de înlănţuire flexibilă a modulelor existente în funcţie de cerinţele utilizatorilor finali;

● asigurarea posibilităţii ca o anumită categorie de utilizatori finali (neinformaticieni) să poată lucra direct cu calculatorul, fără o instruire prealabilă şi fără a fi necesar ca ei să introducă manual comenzi de operare;

● asigurarea posibilităţii ca anumiţi utilizatori finali să se poată autoinstrui, în mod interactiv, pentru a fi în măsură să opereze pe calculator, prin introducerea manuală de comenzi, crescînd astfel eficienţa utilizării sistemului particular PAC;

● livrarea, de către sistem, a unor informaţii de tip „help" atunci sînd utilizatorul intră în derută.

Din enumerarea cerinţelor, rezultă că un monitor de sistem particular PAC include şi un sistem de înlănţuiri flexibil şi extensibil.

Ca şi în cadrul organizării şi reprezentării datelor, cerinţele pe care trebuie să le satisfacă un monitor de sistem particular PAC, ridică probleme cu un grad mare de dificultate, probleme ce nu apar în cadrul aplicaţiilor uzuale, mai ales cele de gestiune economică, cu care am fost obişnuiţi.

Bibliografie

1. ALCOCK, B., *GENESIS Reference manual*, The GENESIS Centre, Loughborough, 1971.
2. ARKINSON, M., WISEMAN, N., *Data management requirements for large scale design and production system*, Reinbow Memo 128, Cambridge Univ., 1976.
3. ARSAC, J., *La conception des programmes*, Pour la science, numéro spécial, no. 85, Nov. (1984).
4. ALFORD, M. W., *A Requirements engineering methodology for realtime processing*, IEEE Trans, on Software Eng., vol. SE-3, no. 1 (1977).
5. ARMITAGE, B. S., HALL, P. A., *Conceptual schema for CAD*, The British Ship Res. Assoc., Wallsend, England, 1976.
6. BALTAC, V., *Industrializarea programării*, Simpozionul dedicat industrializării elaborării de software, Acad. R.S.R., apr. 1983.
7. BANDURSKI, A. E., JEFFERSON, D. K., *Data description for CAD*, Proc. of ACM SIGMOD, San Jose, 1975.
8. BATES, M., VITTAL, J., *Tools for the development of systems for human factors experiments: an example for the SSA*, IEEE Trans. on Systems, Man, and Cybernetics Society, vol. SMC-12, nr. 2 (1982).
9. BATINI, C., *Top-down design in the entity-relationship model*, în: CHEN, P. (ed.), Entity-relationship approach to the systems analysis and design, North-Holland, 1980.
10. BERTRAND, O. P., *Le développment d'applications interactives*, IBM France, Mai 1979.
11. BLAIN, G., LABARTHE, A., RAULT, J. C., *An aid to scientific programming based upon a „Bank of algorithms"*, Proc. of IFIP Congress, 1974.
12. CARROLL, J. M., THOMAS, J. C., *Metaphor and the cognitive representation of computing systems*, IEEE Trans. on Syst., Man, and Cybernetics Society, vol. SMC-12, nr. 2 (1982).
13. CRIŞAN, I., *Tehnologia ca sistem*, Ed. şt. şi encicloped., 1980.
14. DAM VAN, A., *Les logiciels graphics*, Pour la science, numéro spécial, no. 85, Nov. (1984).
15. DAVID, B., *Methodologie pour la construction de systemes CAO*, L'INP Grenoble, 1981.
16. DAVIDOVICIU, A., *Specificul elaborării industriale a programelor pentru aplicaţii de sisteme informatice industriale distribuite şi în timp real*, BRI, nr. 4 şi 5 (1983).
17. DONEZEAU-GOUGE, V., *Programming environments based on structured editors*, INRIA, Rapport technique, nr. 5, Oct. 1981.
18. DRĂGĂNESCU, M., *Industria informaţiei şi programele informatice*, BRI, nr. 3 (1983).
19. ECARNACAO, J., SCHLECHTENDAHL, E., *Computer Aided Design — Fundamentals and System Architecture*, Springer Verlag, 1983.

20. ENDERLE, G., SCHLECHTENDAHL, E., *The design of the integrated* CAD systems REGENT, Proc. Workshop on General Purpose CAD Syst., Toulouse, France, Dec. 1974.
21. ENDERLE, G., SCHLECHTENDAHL, E., *The CAD system REGENT*, 12th Design Aut. Conf., Boston, Mass., June 1975.
22. ENDERLE, G., KANSY, K., PFAFF, G., *Computer Graphics Programming, GKS — The Graphics Standard*, Springer Verlag, 1984.
23. FENCHEL, R.S., ESTRIN, G., *Self-describing systems using integral help*, IEEE Tran. on Syst., Man. and Cybernetics Society, vol. SMC-12, nr. 2 (1982).
24. FENVES, S. J., *Integration of database management and project contol for engineering design*, Proc. on Database for Interact. Design, Waterllo, 1975.
25. FOISSEAU, M. J., *Proposition pour un formalisme conceptuel et pour une architecture de systemé général de CAO s'appuyant sur un SGBD*, Contrat INRIA nr. 77.157, 1979.
26. FOLEY, D. J., VAN DAM, A., *Fundamentals of interactive computer graphics*, Addison-Wesley, 1983.
27. GEORGESCU-ROENGEN, N., *Entropia și procesele economice*, Ed. politică, 1978.
28. GILOL, W. K., *Interactive Computer Graphics*, Prentice-Hall, 1982.
29. GREENSTEIN, J. S., ROUSE, W. B., *A model of human decisionmaking in multiple process monitoring situations*, IEEE Trans. on Systems, Man, and Cybernetics Society, vol. SMC-12, nr. 2 (1982).
30. HOC, M. J., *Analysis of beginners' problem-solving strategies in programming*, Cognitive Engineering, Vrije Univ., Amsterdam, aug. 1982.
31. JORDAN, J. C., *ICES Programmers reference manual*, MIT, 1967.
32. JACOB, R. J. K., *Designing a human-computer interface with software specification techniques*, Proc. of the Second Symp. on Empirical Foundations of Information and Software Scien., Atlanta, 1984.
33. KETIL, BO, *A text analyzer generator for interactive command languages in CAD*, în: Allan, J. J., (ed) CAD Systems, North-Holland, 1977.
34. LEINEMAN, K., SCHLECHTENDAHL, E. G., *The REGENT system for CAD*, în: Allan, J. J., op. cit.
35. LESK, M., *Les logiciels de gestion de l'information*, Pour la science, numéro spécial, no. 85, Nov. (1984).
36. LINDEN, C. A., *Grammars which describe large bodies of data CAD*, CAD, nr. 1 (1978).
37. LOGCHER, R. D., CONNER, J. J., NELSON, M. F., *ICES STRUDL II, Engineering user's manual*, MIT, 1971,
38. LOPEZ, L. A., *POLO, Problem Oriented Language Organizer*, Computers and Structures, nr. 2 (1972).
39. MANOLESCU, G., *A generalized CAD prototype system*, Proc. of Simulation and Modelling Symp., ASME, Athena, 1984.
40. MANOLESCU, G., *Un sistem prototip generalizat pentru CPAC*, ICI, 1983.
41. MANOLESCU, G., *Ingineria concepției: Metoda deciziilor descendent structurate*, Probleme de automatizări, vol. 12 (1982).
42. MANOLESCU, G., CONSTANTINESCU R., LEPADATU, A., LAPADATU, C., TUDOR, A., *A biparty grammar as a tool for defining a manmachine dialogue*, Proc. of IFIP Symp. „Network in Office Automation", Sofia, 1984.
43. MANOLESCU, G., *CAD în contextul apariției calculatoarelor de generația a V-a*, Simp. asupra calc. de gen. a V-a, Acad. R.S.R., 1984.
44. MANOLESCU, G., *Abordarea ierarhic structurată și informatica*, Ed. Acad., 1982.
45. MARTIN, J., *Design of man-computer dialogues*, Prentice-Hall, 1979.
46. MARCUS, R. S., *User assistance in bibliographic retrieval networks through a computer intermediary*, IEEE Trans. on Systems, Man, and Cybernetics Society, vol. SMC-12, nr. 2 (1982).
47. MEDINA-MORA, R., *An incremental programming environment*, IEEE Trans. on Soft. Eng., vol. SE-7, nr. 5 (1981).
48. MELENCIUC, F., *SET, versiunea 2, documentare*, ICI, 1985.
49. NAKAMURA, K., *Design of a question-answeing system for database interfaces using similarity knowledge bases*, Ph. D. disertation, Kyoto University, 1983.

50. PRICE, L., *Thumb: An interactive tool for accessing and maintaining text*, IEEE Trans. on Systems, Man, and Cybernetics Society, vol. SMC-12, nr. 2 (1982).
51. PRICHARD, M. A., *PSU-ICES, Implementation features and philosophy of the PSU-ICES*, CAD Lab. Report Nr. 71-3, MIT, 1971.
52. RAMAMOORTHY, G. S., *A design methodology for user oriented computer systems*, Nat. Comp. Conf., AFIPS, 1978.
53. REISNER, P., *Formal grammars as a tool for analyzing ease use: some fundamenta concepts*, Research Report, IBM, San Jose, 95193, 1982.
54. ROBERTS, R., *HELP: A question answering system*, Proc. Fall Joint Conf. Arlington, 1978.
55. ROLLAND, C., *Information system design and CAD*, Euro IFIP 79, North-Holland, 1979.
56. ROOS, D., ICES system-general description, MIT, 1967.
57. ROOS, D., ICES system design, MIT-Press, 1967.
58. SATA, T., WARMAN, E. (eds.), *Man-machine communication in CAD/CAM*, Proc. of IFIP Working Conf., Tokyo, North-Holland, 1980.
59. SCHLECHTENDAHL, E. G., *Comparison of integrated systems for CAD*, Int. Conf. CAD, Southampton, 1974.
60. SUTHERLAND, I., *SKETCHPAD: A man-machine graphical communication system*, AFIPS, FJCC 23 (1963).
61. SHNEIDER, H. J., *Techniques anf formal tools for design, realization and evaluation of evolutionary information system*, în: Hawgood, J. (ed), Evolutionary information systems, North-Holland, 1982.
62. SHNEIDERMAN, B., *Human engineering management plan for interactive systems*, Proc. COMPCON '83, Washington, 1983.
63. SHNEIDERMAN, B., *Correct, Complete operations and other principles of interaction*, in: Salvendy, G. (ed.), Human-computer interaction, Elsevier, 1984.
64. SHNEIDERMAN, B., *Multiparty grammars and related features for defining interactive systems*, IEEE Trans. on Systems, Man, and Cybernetic Society, vol. SMC-12, nr. 2 (1982).
65. SHAPIRO, E., *PASES: A programming environment for PASCAL*, SIGPLAN Notices, nr. 8 (1981).
66. SIME, M. E., *Psychological evaluation of two conditional constructions used in computer languages*, Int. J. of Man-Machine Stud., nr. 5 (1973).
67. SONDHEIMER, N. K., RELLES, N., *Human factors and user assistance in interactive computing system: An introduction*, IEEE Trans. on Systems, Man, and Cybernetics Society, vol. SMC-12, nr. 2 (1982).
68. SPECTOR, A., *Les logiciels de commands de processus*, Pour la science, numéro spécial, no. 85, Nov. (1984).
69. TEICHOREW, D., *Problem statement analysis: Requirements for the problem statement analyzer (PSA)*, in: Couger, J. D., Knapp, R. W., (eds.), System analysis techniques, John Willey, 1975.
70. TEITELBAUM, T., *The Cornell program synthesizer: a sintax directed programming environment*, Com. of ACM, nr. 9 (1981).
71. TESLER, L., *Les languages de programmation*, Pour la science, numéro spécial, no. 85, Nov. (1984).
72. TRELEAVEN, P. C., LIMA, I. G., *Japan's fifth-generation computer systems*, Computer, Aug. (1982).
73. VALLE, G., *Relational data handling techniques in CAD procedures*, in: Allan, J. J. (ed.), op. cit.
74. WARMAN, A. E., *The possibility for the automatic production of command languaguages*, in: Allan, J. J. (ed.), op. cit.
75. WINOGRAD, T., *Les logiciels de traitment des languages naturelles*, Pour la science, numéro spécial, no. 85, Nov. (1984).
76. WINSTON, P., *Inteligența artificială*, Ed. tehnică, 1981.
77. WIRTH, N., *Structures de données et algorithmes*, Pour la science, numéro spécial, no. 85, Nov. (1984).
78. * * * *Proiect SET*, BRI, nr. 2 (1982).
79. * * * *L'informatique aujourd'hui*, Le Monde dossiers et documents, Sept. (1982).
80. * * * PHIGS, standard ANSI '84.

CASAD — PACHET INTERACTIV PENTRU ANALIZA ȘI PROIECTAREA ASISTATĂ DE CALCULATOR A SISTEMELOR AUTOMATE

dr. ing. A. Varga
dr. ing. A. Davidoviciu

ITCI

1. Introducere

În ultimii ani s-au obținut dezvoltări majore în elaborarea unor algoritmi eficienți și siguri pentru majoritatea problemelor de calcul asociate teoriei moderne a reglării multivariabile [1]—[8]. Existența unor pachete de programe de algebră liniară de înaltă performanță cum sînt LINPACK [9] și EISPACK [10], [11], au contribuit decisiv la implementarea robustă a acestor algoritmi. Recent, s-au elaborat două pachete puternice de subprograme FORTRAN portabile, BIMAS [12] și BIMASC [13] destinate proiectării asistate de calculator a sistemelor automate (PACSA). BIMAS este destinat rezolvării problemelor matematice de bază ale PACSA. BIMASC extinde BIMAS cu subprograme destinate rezolvării problemelor de calcul specifice teoriei moderne a reglării multivariabile. BIMAS și BIMASC implementează cele mai noi dezvoltări în domeniul algoritmilor numerici folosind software-ul numeric cel mai performant existent la această oră. Aceste pachete formează o bază puternică de programe performante ce acoperă majoritatea problemelor de calcul din PACSA.

CASAD (Computer Aided Systems Analysis and Design) este un pachet de programe interactiv bazat pe pachetele BIMAS și BIMASC. CASAD implementează o metodologie completă pentru proiectarea asistată de calculator a sistemelor liniare multivariabile prin metode în spațiul stărilor [2], [14]. Etapele principale ale acestei metodologii constau în: (1) modelarea procesului, (2) analiza sistemului, (3) proiectarea compensatorului și (4) evaluarea performanțelor.

CASAD constă din 15—25 programe (task-uri) selectate în conformitate cu opțiunile utilizatorului. Toate funcțiile implementate se execută prin intermediul unui limbaj de comandă. Fiecare program citește datele de intrare de pe disc și scrie rezultatele pe disc. Datele de ieșire ale fiecărui program se pot folosi ca intrări pentru alte programe ale pachetului. În versiunea actuală, CASAD poate rezolva probleme cu maximum 35—40 variabile de stare, 5 variabile de ieșire, 5 variabile de comandă și 3 perturbații măsurabile. Toate calculele se efectuează în dublă precizie.

CASAD se poate implementa pe familiile de minicalculatoare românești I-100, 102F și CORAL 4011, 4030, sub sistemul de operare MIX. CASAD se poate utiliza în egală măsură și pe familia de minicalculatoare DEC PDP-11 sub sistemul de operare RSX-11M V3.2 sau V4.0.

2. Descrierea pachetului CASAD

2.1. Scopul și caracteristicile pachetului

Scopul principal al elaborării pachetului interactiv CASAD este acela de a dispune pe minicalculator de o colecție puternică de programe destinate modelării, analizei, proiectării și simulării asistate de calculator a sistemelor automate. Programele din CASAD au la bază pachetele BIMAS și BIMASC, care includ și un număr de subprograme din pachetele bine cunoscute EISPACK, LINPACK și ODEPACK. Deși

scopul principal a fost valorificarea maximă a fondului de subprograme din BIMAS și BIMASC, CASAD implementează și algoritmi noi, cum sînt de exemplu pentru acordarea optimală a regulatoarelor de la ieșire de tip P sau PI, cît și pentru simularea eficientă a structurilor de comandă continue ce conduc la ecuații diferențiale „stiff" [8].

Abordarea structurală de rezolvare a problemelor complexe folosită în cadrul pachetelor BIMAS și BIMASC se reflectă și în organizarea pachetului CASAD, atît la nivelul întregului pachet cît și la nivelul programelor de calcul individuale. Datorită modularității pachetelor BIMAS si BIMASC, segmentarea programelor din CASAD s-a realizat relativ ușor. În acest fel programele din CASAD se pot utiliza pentru a rezolva pe minicalculator, în dublă precizie, probleme cu 35—40 variabile de stare.

Toate funcțiunile din CASAD se exploatează prin intermediul unui limbaj de comandă. Linia de comandă, introdusă de la un terminal conține toate informațiile necesare execuției unui program selectat. Opțiunile sau parametrii nespecificați în comandă au valori implicite, astfel încît operarea uzuală cu CASAD este foarte ușoară. În vederea ușurării operării s-au implementat facilități de tip HELP pentru toate comenzile implementate.

Programele din CASAD sînt scrise în FORTRAN, cu excepția unui număr de 5 rutine scrise în limbajul MACRO-11 cu ajutorul cărora se definește sintaxa comenzilor, se preia și se analizează linia de comandă. Cîteva facilități ale sistemului de operare, folosite în cadrul CASAD, contribuie la o utilizare foarte comodă a pachetului.

2.2. Modele de sisteme

Majoritatea calculelor în CASAD se efectuează asupra unor modele de stare liniare constante, continue sau discrete de forma:

$$\lambda x(t)=Ax(t)+B\,u(t)+Ew(t)$$
$$y(t)=Cx(t) \tag{1}$$

unde x este vectorul de stare, u este vectorul de comandă, w este vectorul perturbațiilor măsurabile, y este vectorul ieșirilor măsurate, iar λ este operatorul diferențial d/dt pentru sisteme continue sau operatorul de anticipare $\lambda x(t)=x(t+1)$ pentru sisteme discrete. În (1), A, B, E, C sînt matrici reale.

O altă descriere folosită în CASAD este de tipul intrare-ieșire de forma:

$$Y(\lambda)=G(\lambda)U(\lambda)+G_d(\lambda)W(\lambda) \tag{2}$$

unde Y, U și W sînt transformatele vectorilor de ieșire, comandă și perturbație, respectiv. În cazul continuu se utilizează transformata Laplace, iar în cazul discret transformata Z. În (2), $G(\lambda)$ și $G_d(\lambda)$ sînt matrice de transfer avînd ca elemente funcții de transfer raționale conținînd eventual elemente cu timp mort (numai în cazul continuu).

Utilizarea pachetului CASAD presupune existența unui model de forma (1) sau (2) pentru procesul condus. Acest model se poate obține prin tehnici de modelare și liniarizare, utilizînd principiile de bază ale fizicii și chimiei sau prin identificarea experimentală.

2.3. Structuri de date utilizate în CASAD

Din cele de mai sus, se observă că structura de bază a datelor este matricea, avînd ca elemente numere reale sau funcții de transfer. Vectorii se asimilează cu matrici avînd o linie sau o coloană. CASAD utilizează mai multe tipuri de structuri de date formate din două sau mai multe matrici. De exemplu structura cea mai des folosită este formată din matricile cuadruplului (A, B, C, E) care definește sistemul (1).

În versiunea actuală a CASAD se utilizează 12 structuri de date. Cele mai importante dintre acestea sînt bineînțeles cele pe care utilizatorul are obligația să le

constituie şi acestea sînt: model de stare, model intrare-ieşire, condiţii iniţiale pentru simulare, valori proprii dorite, matrici de ponderare. Alte structuri de date se generează automat de către pachet şi acestea nu sînt de regulă modificate sau create de utilizator. Acestea din urmă conţin: model de estimator, matrici de reacţie, matrici de acţiune-directă, serii de timp obţinute din simulare etc.

Fiecare structură de date este memorată ca un fişier sursă pe discul magnetic. Fiecare fişier conţine pe lîngă datele de bază (matrici), informaţii despre formatul de memorare, numărul de identificare a structurii, dimensiunile matricilor, tipul sistemului (continuu sau discret). Structurile de date deşi sînt predefinite permit o comunicare uşoară între diferite task-uri la nivel de date. Operaţiile de bază referitoare la gestiunea datelor (creare, ştergere, actualizare, listare, copiere) cad în sarcina explicită a utilizatorului şi se fac prin intermediul utilitarelor standard de editare (EDI, EDT, TECO, SLP) şi de operare cu fişiere (PIP), oferite de sistemul de operare al minicalculatorului.

Fişierele se identifică printr-o descriere de fişier de forma:

numefişier. tipfişier; versiune.

Tipul de fişier furnizează un mod convenabil de a discerne între diferite descrieri ale aceluiaşi sistem sau diferite componente ale aceleaşi configuraţii de comandă. De exemplu, modelul de stare standard al unei maşini de hîrtie se poate numi MH.SSM, iar descrierea intrare-ieşire prin matrici de transfer al aceluiaşi sistem se poate numi MH.TRM. Elemente diferite ale configuraţiei compensatorului proiectat cum sînt matrici de reacţie, estimatorul de stare, matrici de acţiune directă („feed-forward"), model de referinţă, se pot denumi ca MH.GAM, MH.ESM, MH.FFM, MH.RFM, respectiv. Utilizarea tipului de fişier în acest mod permite definirea uşoară a unor tipuri implicite pentru datele de intrare şi ieşire specificate în comenzile pachetului CASAD.

2.4. Limbajul de comandă al pachetului

Limbajul de comandă al pachetului CASAD reflectă natura interacţiunii dintre utilizator şi calculator pentru tipul de aplicaţie în cauză. Utilizatorul selectează operaţia ce urmează să o efectueze, specifică descrierile pentru datele de intrare şi ieşire, selectează opţiunile pentru desfăşurarea calculelor în modul dorit. Majoritatea comenzilor din CASAD au forma [18]:

>opc [ieşire] [/sw]=intrare [/sw] [, intr2] [, intr3] ...

unde *opc* specifică codul operaţiei, *intrare, intr 2, intr 3,...* sînt specificatori pentru fişiere de date de intrare, *ieşire* este specificatorul pentru fişierul de date de ieşire, iar */sw* sînt comutatori de intrare sau ieşire. Comutatorii pot fi ataşaţi atît la specificatori de fişier de intrare cît şi la cele de ieşire şi se folosesc pentru precizarea unor parametrii sau a unor opţiuni ale utilizatorului. Entităţile cuprinse între parantezele drepte sînt opţionale. Toate entităţile opţionale au valori implicite. De exemplu, numele de fişier implicit pentru *intr2, intr3,...* este acelaşi cu cel specificat în *intrare*. De asemenea, fiecare descriere de fişier folosită are extensia implicită. Folosirea valorilor implicite contribuie în mare măsură la operarea comodă a pachetului CASAD.

Ca un exemplu să considerăm comanda

>MNE MH/PR=MH/LI/BD/SM:0.7/TOL:1.E-6, POLES

Această comandă se poate folosi pentru proiectarea unui estimator de stare de ordin minimal pentru o maşină de hîrtie. Modelul de stare al maşinii de hîrtie este conţinut în fişierul MH.SSM, iar polii doriţi pentru estimator se găsesc în fişierul POLES.COM. SSM şi COM sînt extensii implicite pentru fişierele de intrare asociate comenzii MNE. Matricile rezultate ale estimatorului de stare se memorează în fişierul MH.ESM. Comutatorii utilizaţi în linia de comandă specifică unele opţiuni ale utilizatorului şi furnizează anumiţi parametrii de proiectare. Astfel, parametrii şi datele de intrare se vor afişa la terminal (/LI), rezultatele obţinute se vor tipări la

imprimantă (/PR), matricea de stare a estimatorului va avea forma bloc-diagonală (/BD), marginea de stabilitate pentru alocarea polilor va fi 0,7 (/CM:0,7), iar toleranța care se va utiliza în testarea observabilității este 10^{-6} (/TOL:1.E-6). Toți comutatorii au valori implicite, iar comutatorii logici cum sînt /LI, /PR, /BD se pot folosi și în forme negate /-LI, /-PR, /-BD.

Pentru a evidenția modul de operare *ușor* permis de utilizarea valorilor implicite, vom utiliza în locul comenzii precedente comanda

>MNE MH=MH

care efectuează în esență aceleași calcule. În acest caz, polii doriți pentru estimator se găsesc în fișierul MH.COM, datele de intrare și de ieșire nu se afișează, matricea de stare a estimatorului va rezulta în forma cuasi-triangulară, marginea de stabilitate folosită va fi —0,7 pentru un sistem continuu și 0,6 pentru un sistem discret, iar toleranța pentru tesarea observabiliății este 10^{-5}.

O facilitate importantă oferită de sistemul de operare al minicalculatorului este posibilitatea de predefinire a unor secvențe de comenzi. Această facilitate *de tip macro* poate fi utilizată eficient în rezolvarea repetată a unor probleme de proiectare complexe sau pentru evitarea introducerii de la terminal a unor linii de comandă lungi. Această facilitate permite de asemenea elaborarea unui mod de lucru conversațional pentru fiecare comandă implementată.

Sintaxele tuturor comenzilor pachetului CASAD sînt prezentate în detaliu în *Manualul de utilizare* al pachetului [16].

3. Funcțiunile pachetului CASAD

CASAD implementează o metodologie completă de proiectare asistată de calculator a sistemelor liniare multivariabile continue sau discrete prin metode în spațiul stărilor. Prima etapă a acestei metodologii este *modelarea sistemului* și are ca scop determinarea unui model convenabil care se poate utiliza în analiza și proiectarea sistemului automat. Programele de modelare conținute în CASAD sînt:

TMCD — discretizarea modelelor intrare-ieșire;
TSCD — discretizarea modelelor de stare;
NMR — construirea modelelor de stare neminimale din modele intrare-ieșire;
MNR — construirea modelelor de stare minimale din modele de stare neminimale;
TSM — determinarea modelelor intrare-ieșire din modele de stare;
TSO — transformări de similaritate ortogonale;
TSI — transformări de similaritate generale;
TCF — calculul formelor standard de controlabilitate-observabilitate;
TBAL — transformări de balansare;
TRED — reducerea ordinului prin balansare internă.

Analiza sistemului permite determinarea unor proprietăți ale modelului (stabilitate, controlabilitate, observabilitate) cît și verificarea unor condiții de existență a soluției unor probleme de proiectare. Programele de analiză conținute în CASAD sînt:

SMA — analiza stabilității, controlabilității/stabilizabilității și observabilității/detectabilității modelelor de stare;
MZE — calculul zerourilor sistemelor multivariabile.

Proiectarea compensatorului presupune determinrea pas cu pas a elementelor acestuia prin metode de proiectare specifice. Structura de compensator cea mai complexă care se poate proiecta cu CASAD conține: regulator robust, model intern, estimator de stare, acțiune directă de la referințe și perturbații (feed-forward) și mo-

del de referință. Sînt prevăzute de asemenea metode de acordare optimală a regulatoarelor PI de la ieșire. Programele de proiectare conținute în CASAD sînt:

PAL — proiectarea regulatoarelor robuste de la stare prin alocarea polilor;
LQN — proiectarea regulatoarelor robuste de la stare prin optimizarea liniar-pătratică;
NME — proiectarea estimatoarelor de stare de ordin complet (neminimale) prin alocarea polilor;
KEN — proiectarea estimatoarelor de stare de tip Kalman;
MNE — proiectarea estimatoarelor de stare de ordin minimal prin alocarea polilor;
OPI — acordarea optimală a regulatoarelor PI de la ieșire;
OIF — acordarea optimală a regulatoarelor integrale de la ieșire;
MFF — proiectarea regulatoarelor cu acțiune directă pentru semnale treaptă, utilizînd modele intrare-ieșire;
SFF — proiectarea regulatoarelor cu acțiune directă pentru semnale treaptă, utilizînd modele de stare.

Evaluarea performanțelor diferitelor configurații de comandă se face prin simulare. Simularea sistemului în buclă deschisă servește în multe cazuri și ca un instrument puternic de analiză a procesului. Se pot în acest mod determina influența intrărilor asupra ieșirilor, natura răspunsului la treaptă sau rampă, timpul de creștere, durata regimului tranzitoriu cît și alte măsuri cantitative sau calitative ale dinamicii sistemului. Programele de simulare conținute în CASAD sînt:

SDS — simularea structurilor de comandă liniare discrete;
SCS — simularea structurilor de comandă liniare continue;
SSCS — simularea structurilor de comandă liniare continue „stiff";
SHS — simularea structurilor de comandă liniare hibride: sistem continuu condus discret;
PLOT — reprezentarea grafică a seriilor de timp obținute prin simulare.

O prezentare mai detaliată a funcțiunilor pachetului este dată în *Manualul de descriere* [15].

4. Proiectarea unui sistem de reglare frecvența — putere de schimb

Exemplul prezentat în această secțiune demonstrează utilizarea pachetului CASAD în proiectarea unui sistem de conducere cu calculator pentru două sisteme energetice interconectate [19]. Sistemul constă din două generatoare electrice conectate prin linii de interconexiune. Scopul reglării este menținerea frecvențelor ambelor grupuri cît și a încărcării liniei de interconexiune la valori constante, în condiții de variații ale sarcinii celor două grupuri. Ecuațiile liniarizate ale sistemului sînt de forma

$$\dot{x}=Ax+Bu+Ew$$
$$y=Cx \tag{3}$$
$$y_r=C_rx$$

unde $x\in R^7$, $u\in R^2$, $w\in R^2$, $y\in R^3$, iar $y_r\in R^2$ sînt ieșirile reglate. Presupunem că ieșirile reglate sînt și măsurabile și satisfac condiția $y_r=[I\ 0]\ y$ (I este matricea unitate 2×2). Variabilele u, w și y au următoarele semnificații fizice:

u_i $(i=1,2)$ — deviația poziției variatorului de turație pentru grupul i;
w_i $(i=1,2)$ — deviația sarcinii grupului i;
y_1 — deviația frecvenței grupului 1;

y_2 — deviaţia puterii de schimb;
y_3 — deviaţia frecvenţei grupului 2.

Toate deviaţiile se consideră faţă de valori de regim nominale.

Matricile sistemului sînt:

$$A=\begin{bmatrix} -0{,}04165 & 0{,}0 & 4{,}92 & -4{,}92 & 0{,}0 & 0{,}0 & 0{,}0 \\ -5{,}21 & -12{,}5 & 0{,}0 & 0{,}0 & 0{,}0 & 0{,}0 & 0{,}0 \\ 0.0 & 3{,}33 & -3{,}33 & 0{,}0 & 0{,}0 & 0{,}0 & 0{,}0 \\ 0{,}545 & 0{,}0 & 0{,}0 & 0{,}0 & -0{,}545 & 0{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 & 0{,}0 & 4{,}92 & -0{,}04165 & 0{,}0 & 4{,}92 \\ 0{,}0 & 0{,}0 & 0{,}0 & 0{,}0 & -5{,}21 & -12{,}5 & 0{,}0 \\ 0{,}0 & 0{,}0 & 0{,}0 & 0{,}0 & 0{,}0 & 3{,}33 & -3{,}33 \end{bmatrix}$$

$$B=\begin{bmatrix} 0{,}0 & 0{,}0 \\ 12{,}5 & 0{,}0 \\ 0{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 \\ 0{,}0 & 12{,}5 \\ 0{,}0 & 0{,}0 \end{bmatrix} \quad E=\begin{bmatrix} -4{,}92 & 0{,}0 \\ 0{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 \\ 0{,}0 & -4{,}92 \\ 0{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 \end{bmatrix} \quad C=\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Aceste matrici se încarcă de către utilizator într-un fişier sursă cu numele SE.SSM utilizînd editorul de texte standard al sistemului de operare.

4.1. Analiza şi modelarea sistemului

Prin intermediul comenzii CASAD

```
>SMA SE
```

se calculează polii sistemului şi se face analiza proprietăţilor acestuia. Polii calculaţi sînt: {—0,5181±3,126; —1,355±2,187; —1,692; —13,144; —13,162}, sistemul este evident stabil, este de asemenea, controlabil şi observabil. Sistemul corespunzător tripletului (A, B, C_r nu are zerouri. Acest lucru s-a determinat cu comanda:

```
>MZE SE
```

Analiza făcută arată că problema de reglare robustă are soluţie [2].

Pentru obţinerea unui compensator de ordin cît mai redus, s-a încercat o reducere a ordinului modelului iniţial. Aproximarea rezultată prin balansare internă a sistemului, obţinută cu comanda

```
>TRED SER=SE/TOL : 1.E-2
```

conduce la un sistem redus de ordin 5 descris de ecuaţiile

$$\begin{aligned} \dot{x} &= \bar{A}\bar{x}+\bar{B}u+\bar{E}w \\ y &= \bar{C}\bar{x} \\ y_r &= \bar{C}_r\bar{x} \end{aligned} \qquad (4)$$

care are matricile

$$\bar{A}=\begin{bmatrix} -0{,}668 & 3{,}201 & 0{,}0 & 0{,}0 & -0{,}897 \\ -2{,}971 & -0{,}165 & 0{,}0 & 0{,}0 & 0{,}147 \\ 0{,}0 & 0{,}0 & -1{,}084 & -2{,}464 & 0{,}0 \\ 0{,}0 & 0{,}0 & 1{,}835 & -1{,}5 & 0{,}0 \\ 0{,}587 & 0{,}358 & 0.0 & 0{,}0 & -1{,}6 \end{bmatrix}$$

$$\bar{B}=\begin{bmatrix} -0,634 & 0,634 \\ -0,723 & 0,723 \\ 0,766 & 0,766 \\ -1,278 & -1,278 \\ 1,012 & -1,012 \end{bmatrix} \qquad \bar{E}=\begin{bmatrix} 1,487 & -1,487 \\ -0,266 & 0,266 \\ -1,467 & -1,467 \\ -0,00125 & -0,00125 \\ -0,144 & 0,144 \end{bmatrix}$$

$$\bar{C}=\begin{bmatrix} -1,612 & 0,609 & 1,655 & 1,278 & -0,938 \\ 0,176 & 0,667 & 0,0 & 0,0 & 0,572 \\ 1,612 & -0,609 & 1,655 & 1,278 & 0,938 \end{bmatrix}$$

Aceste matrici rezultă într-un fişier de date cu numele SER.SSM.

Analiza sistemului redus cu comenzile

>SMA SER

>MZE SER

arată că polii sistemului redus sînt: $\{-0,505\pm3,12;\ -1,292\pm2,116;\ -1,423\}$, iar zerourile sistemului $(\bar{A}, \bar{B}, \bar{C}_r)$ sînt $\{-39,65;\ 13,22;\ 19,54\}$. Se observă că prin reducerea ordinului, zerouri mari au fost introduse în semiplanul drept. Cu toate acestea, problema de reglare are soluţie şi pentru modelul redus.

Conducerea cu calculator presupune utilizarea unui model discretizat de forma

$$\begin{aligned} \bar{x}_d(k+1) &= \bar{A}_d\bar{x}_d(k)+\bar{B}_d u(k)+\bar{E}_d w(k) \\ y(k) &= \bar{C}\bar{x}_d(k) \\ y_r(k) &= \bar{C}_r\bar{x}_d(k) \end{aligned} \qquad (5)$$

Matricile sistemului discretizat, calculate pentru o perioadă de eşantionare de 0,1 secunde, obţinute cu comanda

>TSCD SEDR=SER/DT : 0.1

sînt

$$\bar{A}_d=\begin{bmatrix} 0,888 & 0,3 & 0,0 & 0,0 & -0,0766 \\ -0,28 & 0,938 & 0,0 & 0,0 & 0,0254 \\ 0,0 & 0,0 & 0,877 & -0,215 & 0,0 \\ 0,0 & 0,0 & 0,16 & 0,841 & 0,0 \\ 0,0467 & 0,041 & 0,0 & 0,0 & 0,8502 \end{bmatrix}$$

$$\bar{B}_d=\begin{bmatrix} -0,0755 & 0,0755 \\ -0,0604 & 0,0604 \\ 0,0865 & 0,0865 \\ -0,114 & -0,114 \\ 0,0904 & -0,0904 \end{bmatrix} \qquad \bar{E}_d=\begin{bmatrix} 0,138 & -0,138 \\ -0,0475 & 0,0475 \\ -0,138 & -0,138 \\ -0,0124 & -0,0124 \\ -0,01 & 0,01 \end{bmatrix}$$

Matricile sistemului discretizat se obţin în fişierul SEDR.SSM. Cu comanda

>MZE SEDR

s-au calculat zerourile sistemului discretizat ca fiind $\{-79,64;\ 4,077;\ -0,1884\}$, deci problema de reglare robustă are soluţie pentru sistemul discret.

4.2. Proiectarea compensatorului

Compensatorul robust discret constă dintr-un regulator optimal de la stare, un estimator minimal şi un model intern format din integratoare numerice. Ecuaţiile compensatorului, care se programează pe calculatorul de proces, sînt:

$$\begin{aligned} u(k) &= H_i x_i(k)+Hx_e(k) \\ x_i(k+1) &= x_i(k)+(r(k)-y_r(k)) \\ x_e(k) &= My(k)+Nz(k) \\ z(k+1) &= Fz(k)+Gy(k)+Lu(k) \end{aligned} \qquad (6)$$

unde r este vectorul referinţelor, iar x_i este starea integratoarelor.

Matricile regulatorului, H_i şi H se determină prin minimizarea criteriului de performanţă pătratic

$$J=\sum_{k=0}^{\infty}\left\{x'^T(k)\begin{bmatrix}Q_i & O\\ O & \overline{C}_r^T Q \overline{C}_r\end{bmatrix}x'(k)+u^T(k)Ru(k)\right\}$$

unde x' este starea unui sistem extins format din cuplarea integratoarelor la sistemul discret. Matricile de ponderare au valorile:

$$Q_i=\begin{bmatrix}10 & 0\\ 0 & 10\end{bmatrix},\quad Q=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix},\quad R=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$$

şi se introduc de utilizator în fişierul cu numele SEDR.WEM. Utilizînd comanda

>LQN SEDR=SEDR/IMT : 1

unde comutatorul /IMT : 1 specifică folosirea integratoarelor, se obţin următoarele valori pentru matricile regulatorului:

$$H_i=\begin{bmatrix}1{,}65 & 0{,}3\\ 0{,}462 & -2{,}191\end{bmatrix}$$

$$H=\begin{bmatrix}10{,}79 & 0{,}788 & -11{,}0 & -3{,}278 & 3{,}047\\ -2{,}138 & 9{,}634 & -4{,}05 & -0{,}54 & 3{,}676\end{bmatrix}$$

Aceste valori rezultă în fişierul SEDR.GAM.

Estimatorul de stare, avînd ca ieşire starea estimată x_e, s-a proiectat prin alocarea polilor. Polii alocaţi{0,4; 0,3} se introduc în fişierul SEDR.COM. Utilizînd comanda

>MNE SEDR=SEDR/BD

se determină matricile estimatorului

$$F=\begin{bmatrix}0{,}3 & 0\\ 0 & 0{,}4\end{bmatrix}\quad G=\begin{bmatrix}0{,}409 & 0{,}367 & -0{,}409\\ -0{,}276 & 0{,}0 & -0{,}276\end{bmatrix}\quad L=\begin{bmatrix}-0{,}134 & 0{,}134\\ 0{,}14 & 0{,}14\end{bmatrix}$$

$$M=\begin{bmatrix}-0{,}547 & -0{,}538 & 0{,}547\\ -0{,}132 & 0{,}544 & 0{,}132\\ 0{,}521 & 0{,}0 & 0{,}521\\ -0{,}283 & 0{,}0 & -0{,}283\\ 0{,}322 & 1{,}279 & -0{,}322\end{bmatrix}\quad N=\begin{bmatrix}0{,}57 & 0{,}0\\ 0{,}443 & 0{,}0\\ 0{,}0 & 0{,}611\\ 0{,}0 & -0{,}791\\ -0{,}691 & 0{,}0\end{bmatrix}$$

care se depun în fişierul SEDR.ESM. Prin utilizarea comutatorului /BD, matricea F rezultată este în formă diagonală.

4.3. Evaluarea performanţelor

Simularea sistemului hibrid format din sistemul continuu în buclă deschisă (3) comandat discret prin compensatorul (6) s-a efectuat cu comanda

>SHS SEDR=SE/TI : 0 : 3 : 0.1/SFC, SE, SEDR, SEDR

în care comutatorul /TI specifică timpul iniţial (=0), timpul final (=3) şi perioada de eşantionare (0,1), iar comutatorul /SFC specifică utilizarea unui compensator format dintr-un regulator de stare şi estimator. Rezultatele se obţin în fişierul SEDR.TIS şi se pot afişa cu comanda

>PLOT SEDR

În figurile 1 şi 2 se arată variaţiile frecvenţelor grupurilor 1 şi 2, respectiv, pentru o perturbaţie de sarcină de 0,1 p.u. în primul sistem. Comenzile corespunză-

toare generate de compensator sînt prezentate în figurile 3 și 4. Aceste grafice arată că sistemul de comandă rezultat are performanțe satisfăcătoare și în comparație cu lucrarea [19], compensatorul are un ordin dinamic mult mai mic.

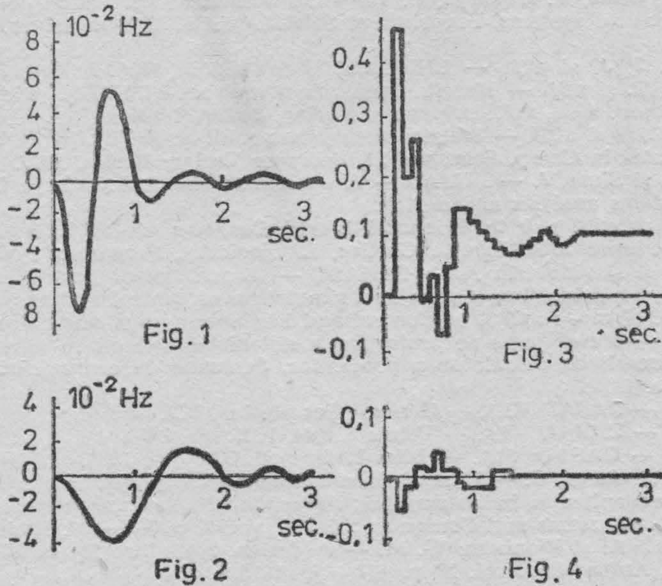


Fig. 1 Fig. 3 Fig. 2 Fig. 4

I. Concluzii

Acest articol a prezentat un pachet de programe performant CASAD, destinat analizei și proiectării asistate de calculator a sistemelor automate. În comparație cu pachete similare existente pe plan mondial, CASAD este printre cele mai avansate din punct de vedere al algoritmilor utilizați și oferă unele facilități care nu sînt implementate în nici unul din pachetele cunoscute. Exemplul numeric prezentat demonstrează utilitatea acestui pachet în rezolvarea unor probleme complexe din practică utilizînd o abordare teoretică modernă.

5. Bibliografie

[1] Van Dooren, P., — The generalized eigenstructure problem in linear system theory, IEEE Trans. Autom. Control, vol. AC-26, p. 111—129, 1981.

[2] Varga, A. — Contribuții la conducerea robustă cu calculator a proceselor continue. Teză de doctorat, I. P. București, 1981.

[3] Varga, A. — A numerically stable algorithm for standard controllability form determination. Electronics Letters, vol. 17, p. 74—75, 1981.

[4] Van Loan, C. F., — Computing matrix integrals involving matrix exponentials. IEEE Trans. Autom. Control, vol. AC-23, p. 395—404, 1978.

[5] Varga, A. și Sima, V. — A numerically stable algorithm for transfer function matrix evaluation. Int. J. Control, vol. 33, nr. 6, 1981.

[6] Varga, A. — A Schur method for pole assignment. IEEE Trans. Autom. Contr., vol. AC-26, p. 517—519, 1981.
[7] Varga, A. — Computer aided design of robust compensators by pole assignment, Preprints of SOCOCO'82 Symp., Sept. 1982, Madrid.
[8] Varga, A., Sima, V. şi Varga, C. V. — On numerical simulation of linear continuous control systems, Preprints of SIMULATION'83 Symposium, Prague, June 1983.
[9] Dongarra, J. J. şi alţii — LINPACK User's Guide, SIAM, Philadelphia, 1979.
[10] Smith, B. T. şi alţii — Matrix eigensystem routines-EISPACK guide, Lect. Notes in Comp. Scie., vol. 6, Springer Verlag, Berlin, 1974.
[11] Garbow, B. S. şi alţii, — Matrix eigensystem routines-EISPACK guide extension, Lect. Notes in Comp. Scie., vol 51, Springer Verlag, Berlin, 1977.
[12] Varga A., şi Sima V. — BIMAS — A basic mathematical package for computer aided systems analysis and design.
Proceedings of the 9 th Triennial World Congress of IFAC, Budapest, Hungary, 2—6 iulie 1984, Eds. J. Gertler, L. Keviczky, Pergamon Press, 1985.
[13] Varga A. şi Davidoviciu A. — BIMASC — A package of Fortran subprograms for analysis, modelling, design and simulation of control systems, Preprints of 3rd IFAC Symp. on CAD in Control and Engineering Systems, Copenhaga, 1985.
[14] Varga A. — Metodologie de proiectare a sistemelor de reglare robuste în vederea conducerii cu calculator a proceselor, Sesiunea de comunicări ICI, Bucureşti, dec. 1979.
[15] Varga, A. — CASAD V1.1. — General Description, ICI, Dec. 1984.
[16] Varga, A. — CASAD V1.1. — User's Guide, ICI, dec. 1984.
[17] Varga, A. — CASAD V1.1. — Mini-Reference, ICI, dec, 1984.
[18] Varga, A. şi Varga, C. — Limbaj de comandă pentru proiectarea asistată de calculator cu aplicaţie în automatică, Preprints of 5th International Conference on Control Systems and Computer Science, vol. 3, p. 303—307, Iunie 1983.
[19] Sebakhy, O. A. şi Wonham, W. M. — A design procedure for multivariable regulators, Automatica, vol. 12, p. 467—478, 1976.

MICROCALCULATORUL PERSONAL-PROFESIONAL FELIX-PC.
Structura şi arhitectură

Prof. dr. ing. *Adrian Petrescu*, şi dr. ing. *Trandafir Moisa*, şi dr. ing. *Nicolae Ţăpuş* şi dr. ing. *Irina Athanasiu* — Institutul Politehnic Bucureşti în colaborare cu ing. *Florea Tănase*, ing. *Dorin Mihu*, ing. *Tudorel Domocoş*, ing. *Alexandru Anghel*, ing. *Gabi Drăghicescu*, ing. *Constantin Alupului*; ing. *Anca Negulescu* — Întreprinderea de Calculatoare Electronice.

1. Introducere

Microsistemul FELIX-PC este un nou tip de microcalculator personal — profesional, bazat pe microprocesoare din generaţia a III-a, cu un grad de integrare tehnologică ridicat, structura compactă şi un sistem de programe ce acoperă o gamă largă de aplicaţii.

Are o structură configurabilă pe 8 sau 16 biţi în funcţie de microprocesorul care se utilizează. Prin structura sa modulară poate fi utilizat ca sistem universal sau dedicat funcţional în aplicaţii specializate. Sistemul poate fi completat cu extensii hardware ce pot acoperi o gamă largă de aplicaţii cu complexitate ridicată.

2. Structura microsistemului

FELIX-PC are o structură modulară, fiind alcătuit dintr-un modul de bază şi module de extensie.

Modulul de bază cuprinde resursele hardware care asigură funcţionarea sa ca sistem universal, cu o configuraţie redusă care include: unitatea centrală, tastatura, consola serială, imprimanta, discuri flexibile.

Modulele de extensie sînt opţionale, fiind utilizate în alcătuirea unor configuraţii adecvate aplicaţiilor sau pentru mărirea disponibilităţii şi facilităţilor sistemului.

Modulul de bază conţine următoarele resurse:

- unitatea de prelucrare bazată pe microprocesoarele 8088/8066 şi 8087;
- memorie RAM de 256 Kocteţi;
- memorie EPROM de 8 Kocteţi — 64 Kocteţi;
- cuplor pentru discuri flexibile de 5¼" sau 8";
- interfeţe pentru:
  - testatură;
  - imprimantă serială;
  - comunicaţie asincronă/sincronă;
  - casetă magnetică audio;
  - generator de tonuri;
- ceas de timp real;
- numărătoare programabile;
- sistem de întreruperi;
- canale de acces direct la memorie;
- conectori pentru module de extensie;
- conectori pentru periferice.

Sistemul FELIX-PC are o magistrală de date configurabilă pe 16 biţi sau 8 biţi în funcţie de microprocesorul care se utilizează: 8086 sau 8088.

Cele două microprocesoare sînt compatibile la nivel de pini şi la nivel de set de instrucţiuni, numai că microprocesorul 8088 comunică cu exteriorul printr-o magistrală de date de 8 biţi iar 8086 comunică cu exteriorul printr-o magistrală de date de 16 biţi. (8088 execută operaţiile de transfer cu exteriorul pe cuvînt în două cicluri maşină, iar 8086 într-un singur ciclu).

Microprocesoarele pot să adreseze 1 Moctet de memorie, avînd în afară de instrucţiunile comune altor microprocesoare şi instrucţiuni de lucru pe şiruri de caractere, instrucţiuni de înmulţire şi împărţire în virgulă fixă şi diverse moduri de adresare.

Frecvenţa de lucru este de 5MHZ iar un ciclu maşină este de 800ns (ciclu de I/E de 1μs).

Microprocesorul este utilizat în modul maxim pentru a putea fi cuplat cu coprocesorul matematic 8087.

Sistemul FELIX-PC a fost prevăzut cu posibilitatea introducerii microprocesorului matematic NDP 8087. Acesta creşte considerabil viteza de lucru (este de circa 100 ori mai rapid decît 8086/8088) în cazul prelucrărilor numerice care cuprind:

— adunarea în virgulă fixă şi mobilă;
— scăderea în virgulă fixă şi mobilă;
— înmulţirea în virgulă fixă şi mobilă;
— împărţirea în virgulă fixă şi mobilă;
— rădăcina pătrată;
— rotunjire;
— funcţii transcendente ca: tangenta, arctangenta (parţiale), logaritm, ridicare la putere;
— etc.

Reprezentarea datelor în virgulă mobilă este conform standardului IEEE.

Din punct de vedere software nu toate translatoarele utilizează în mod implicit coprocesorul 8087 şi, din acest motiv, utilizatorul trebuie să-şi realizeze procedurile corespunzătoare pentru apelarea funcţiilor matematice. Versiunile mai noi de interpretoare şi translatoare pot lucra implicit, printr-o configurare corespunzătoare, cu coprocesorul 8087.

Memoria RAM este realizată cu componente de memorie dinamică 4164 (64K×1) şi are o capacitate de 256 Kocteţi. Este organizată pe 8 sau 16 biţi, fiind prevăzută cu bit de paritate la nivel de octet.

Memoria EPROM, de capacitate între 8 Kocteţi şi 64 Kocteţi, în funcţie de tipul de componente care se utilizează, conţine BIOS-ul care este format din:

— testul resurselor hardware standard;
— rutinele de I/E asociate echipamentelor standard;
— generatorul de caractere pentru modul grafic;
— încărcătorul pentru discul flexibil.

În configuraţia standard, cuplorul pentru discuri flexibile este configurat pentru discuri de 5 1/4".

Discurile flexibile sînt sectorizate software. Cuplorul de disc permite lucrul cu discuri dublă densitate, simplă sau dublă faţă şi utilizează codificarea MFM (modulare în frecvenţă modificată). Discul flexibil este organizat pe 40 piste, cu 9 sectoare de 512 octeţi pe fiecare pistă. Deci capacitatea totală a unui astfel de disc este de 184 320 de octeţi pentru discurile simplă faţă, respectiv 368 640 de octeţi pentru discurile dublă faţă. Rata de transfer este de 250 Kbiţi pe secundă.

Configuraţia de bază conţine şi o interfaţă paralelă programabilă prin intermediul căreia se cuplează:

— interfaţa de tastatură care asigură cuplarea serială a tastaturii. Tastatura generează un număr de 83×2 coduri, denumite coduri de scanare. Pentru fiecare tastă

se generează un cod la apăsare și unul la ridicare. În acest mod se poate determina durata de acționare a unei taste. Tastatura este prevăzută cu un număr mare de taste funcționale.

— comutatoarele de configurare sistem care specifică: capacitatea memoriei RAM, tipul terminalului grafic utilizat, modul de lucru al terminalului grafic, numărul de unități de disc atașate sistemului.

— difuzorul, al cărui control se realizează în două moduri:

— direct prin program, prin poziționarea unui bit de date, generîndu-se un tren de impulsuri;

— prin programarea canalului 2 al contorului programabil, generîndu-se frecvența dorită.

Ambele moduri pot să fie controlate simultan. Difuzorul este utilizat pentru semnalizarea erorilor în etapa de testare și pentru generarea de semnale audio.

— interfața de casetă magnetică audio care asigura o rată de transfer de 1 Kbauds—2 Kbauds. Transferul de date cu caseta, generează și verifică CRC.

Sistemul este prevăzut cu 6 contoare programabile utilizate după cum urmează:

— canalul 0 — generează semnale periodice care asigură cuanta de bază pentru ceasul de timp real al sistemului;

— canalul 1 — generează cereri de cicluri DMA pentru operația de reîmprospătare a memoriei dinamice RAM;

— canalul 2 — generator de tonuri pentru difuzor;

— canalul 3, 4 — generează semnale de tact cu o frecvență stabilită de rata de transfer la recepție transmisie pentru interfața seriala duală.

Sistemul de întreruperi este organizat pe 8 nivele prioritare și este realizat cu 8259. Trei nivele de întrerupere sînt utilizate în cadrul modulului de bază, celelalte fiind disponibile pentru modulele de extensie.

Nivelul 0,prioritate maximă, este alocat canalului 0 al contorului programabil care furnizează o întrerupere periodică utilizată pentru ceasul de timp real al sistemului. Nivelul 1 este alocat tastaturii. Nivelul 6 este alocat discului flexibil.

Cele 8 nivele sînt puse în corespondență cu nivelele de întrerupere 8—15 ale microprocesorului.

Configurația de bază a sistemului include 4 canale de acces direct la memorie Canalul DMA 0 este utilizat pentru operațiile de reimprospătare a memoriei RAM dinamice de pe placa de bază. Cererile de reîmprospătare sînt lansate la fiecare 15 µs de către contorul 1. În acest fel numai 7% din timpul sistemului este consumat cu operația de reîmprospătare. Canalul DMA 2 este utilizat pentru operațiile cu discul flexibil. Canalele DMA 1 și 3 sînt disponibile la conectorii de extensie.

Modulul de bază este structurat în jurul a doua magistrale:

— magistrala sistemului care permite cuplarea extensiilor;

— magistrala locală care cuplează resursele locale ale sistemului.

Descrierea PMS a modulului de baza (fig. 1), prezintă structura sistemului unde s-a considerat:

Pc — procesor central care conține microprocesorul 8086 sau 8088;
Pm — procesor matematic care conține microproceroul 8087;
Ks — unitate de control sistem;
Kpl — unitate de control resurse locale;
Kmlr — unitate de control acces direct la memorie;

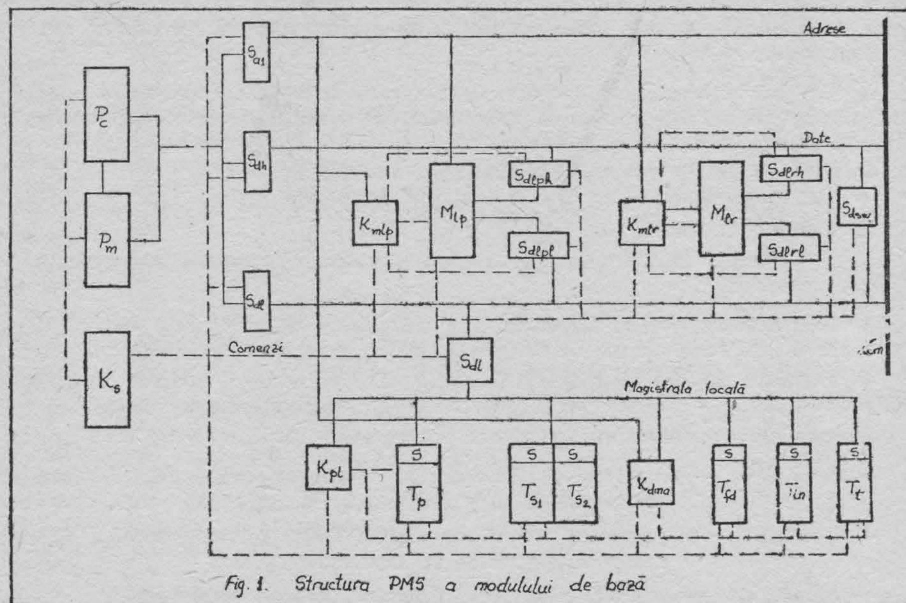


Fig. 1. Structura PM5 a modulului de bază

Kmlr — unitate de control memorie locală RAM;
Kmlp — unitate de control memorie locală EPROM;
Sdi — comutatoare de conectare date la magistrală;
Sai — comutatoare de conectare adrese la magistrală;
Sdsw — comutatoare de configurare magistrală pe 8 sau 16 biți;
Mlp — memorie locală EPROM;
Mlr — memorie locală RAM;
Tp — interfața paralelă programabilă;
Tfd — cuplor de disc flexibil;
Tsi — interfețe seriale sincrone/asincrone programabile;
Tin — sistem de întreruperi;
Tt — generator de tact programabil.

Pentru mărirea disponibilității și flexibilității sistemului, pe placa de bază au fost prevăzuți 4 conectori cu 31 de contacte și 4 conectori dubli cu 31+10 contacte care asigură semnalele de date, adrese și comenzi necesare interfațării modulelor de extensie.

Conectorii de extensie conțin:

— magistrala bidirecționala de date de 16 biți/8 biți;
— magistrala de adrese de 20 biți
— 8 nivele de întrerupere;
— 3 canale DMA;
— linii de control al operației de reîmprospătare a memoriei;
— linii de sincronizare și control.

Conectorii de extensie simpli sînt compatibili, cu magistrala IBM.

Structura magistralei, disponibilă pentru modulele de extensie este următoarea:

GND	B1	A1	I/OCHCK
RESET DRV	B2	A2	D7
+5V	B3	A3	D6
IRQ2	B4	A4	D5
—5VDC	B5	A5	D4
DRQ2	B6	A6	D3
—12V	B7	A7	D2
rezervat	B8	A8	D1
+12V	B9	A9	D0
GND	B10	A10	I/OCHRDY
MEMW	B11	A11	AEN
MEMR	B12	A12	ADR19
IOW	B13	A13	ADR18
IOR	B14	A14	ADR17
DACK3	B15	A15	ADR16
DRQ3	B16	A16	ADR15
DACK1	B17	A17	ADR14
DRQ1	B18	A18	ADR13
DACK0	B19	A19	ADR12
CLOCK	B20	A20	ADR11
IRQ7	B21	A21	ADR10
IRQ6	B22	A22	ADR9
IRQ5	B23	A23	ADR8
IRQ4	B24	A24	ADR7
IRQ3	B25	A25	ADR6
DACK2	B26	A26	ADR5
T/C	B27	A27	ADR4
ALE	B28	A28	ADR3
+5V	B29	A29	ADR2
OSC	B30	A30	ADR1
GND	B31	A31	ADR0

D8	B41	A41	BHE
D9	B42	A42	INTA
D10	B43	A43	EXTHOLDACK
D11	B44	A44	DMAHOLDRQST
D12	B45	A45	PCLK
D13	B46	A46	IRQ1
D14	B47	A47	IRQ0
D15	B48	A48	NMI
	B49	A49	I/OCHOLDRQST
	B50	A50	

Semnalele de pe magistrală au următoarea semnificație:

OSC: (B30) — este un semnal de tact cu perioada de 70 ns cu factor de umplere 50%;

CLK: (B20) — este un semnal de tact cu perioada de 210 ns cu factor de umplere 33%;

RESET DRV:(B2) — este un semnal de inițializare sistem, generat la punerea sub tensiune. Este activ pe „1" și este sincronizat cu frontul descrescător al semnalului de tact;

ADR0—ADR19: (A31—A12) — sînt liniile de adrese. Aceste semnale sînt utilizate pentru adresarea memoriei și a dispozitivelor de I/E. ADR0 este bitul de adresă cel mai puțin semnificativ iar ADR19 este bitul de adresă cel mai semnificativ. Sînt active pe „1" și sînt controlate de unitatea centrala de prelucrare (8086/8078) sau de modulul de acces direct la memorie DMA.

D0—D15: (A9—A2, B41—B48) sînt linii de date. Aceste linii constituie magistrala bidirecțională de date pentru microprocesor, memorie și dispozitivele de I/E. D0 este bitul cel mai puțin semnificativ, iar D15 este bitul cel mai semnificativ.

ALE: (B28) — este un semnal generat de Ks în timpul unui ciclu de acces pe magistrală pentru a indica prezența unor adrese stabile pe magistrala internă a sistemului, constituind un semnal de strob, pe front descrescător, pentru registrele de adrese. Este utilizat impreună cu AEN pentru a indica interfețelor de I/E prezența stabilă a adreselor de la procesor.

AEN: (A11) — este un semnal, activ pe „0", care indică faptul că procesoarele 8086/8088 sau 8087 au controlul magistralei sistemului. Cînd AEN=„1" modulul DMA are controlul asupra magistralei sistemului (date, adrese, comenzi).

I/OCHK: (A1) — este un semnal activ pe „0" care generează o cerere de întrerupere nemascabilă. Este utilizat pentru a anunța erori de paritate la extensia de memorie sau la interfețele de I/E.

I/OCHRDY: (A10) — este o linie de sincronizare cu modulele lente. Are ca efect includerea unor stări de așteptare pînă cînd dispozitivul mai lent răspunde la comanda care i-a fost adresată. Pentru a nu intra în conflict cu procesul de reînprospătare a memoriei, această linie nu trebuie să fie ținută la „0" mai mult de 1 sau 2µs (mai mult de 10 perioade de tact de 210 ns).

MEMR: (B12) — este un semnal care indică o operație de citire din memorie, operație inițiată de procesor sau de DMA. Semnalul este activ pe „0".

MEMW: (B11) — este un semnal care indică o operație de scriere în memorie inițiată de procesor sau de DMA. Semnalul este activ pe „0".

IOR: (B14) — este un semnal care specifică o operație de citire de la un port de intrare. Este generat de procesor sau de DMA și este activ pe „0".

IOW: (B13) — este un semnal care specifică o operație de scriere la un port de ieșire. Este generat de procesor sau de DMA și este activ pe „0".

IRQ0—IRQ7: (A46, A47, B4, B25—B21) — sînt semnale ce specifică cereri de întrerupere de la echipamente externe. Cererea IRQ0 este cea mai prioritară iar IRQ7 este cea mai puțin prioritară. O cerere de întrerupere este generată de trecerea din „0" în „1" a liniei IRQi. Cererea IRQi trebuie să fie menținută în „1" pînă cînd este luata în considerare (se ajunge în rutina de tratare a a întreruperii). IRQ0, IRQ1, IRQ6 sînt utilizate de către modulul de bază, dar pot să fie utilizate și de către un procesor cuplat pe magistrala de extensie.

DRQ1—DRQ3: (B18, B6, B16) — sînt semnale de cerere de acces direct la memorie, prin intermediul modulului DMA. DRQ1 are prioritatea cea mai mare iar DRQ3 are prioritatea cea mai mică. O cerere DRQi este activă pe „1" și trebuie menținută pînă cînd linia de confirmare corespunzătoare, DACKi, devine activă. Linia DRQ2 este utilizată de adaporul de disc flexibil care este inclus în modulul de bază, dar a fost trecută și pe magistrala de extensie pentru compatibilizare cu magistrala sistemului IBM-PC.

DACK0—DACK3: (B19, B17, B26, B15) — sînt semnale de confirmare a luării în considerare a cererilor DRQi, de acces direct la memorie. Sînt active pe „0". Canalul 0 al modulului de acces direct la memorie este utilizat pentru reîmprospătarea memoriei dinamice RAM. În consecință DACK0 este o indicare a prezenței unui acces DMA pentru reîmprospătarea memoriei dinamice.

T/C: (B27) — este un semnal care indică faptul că unul dintre contoarele modulului de acces direct la memorie a ajuns la zero. Acest semnal este activ pe „1".

$\overline{\text{I/OCHOLDRQST}}$: (B49) — este un semnal prin care interfețele de I/E cer suspendarea activității unității centrale de prelucrare, pentru a avea acces la controlul magistralei sistemului.

DMAHOLDRQST: (A44) — este un semnal prin care DMA cere suspendarea activității unității de prelucrare, pentru a avea acces la controlul magistralei sistemului. Este un semnal generat de modulul de bază și reprezintă o informație de stare pentru modulele de extensie.

EXTHOLDACK: (A43) — este un semnal prin care se anunță faptul că un procesor conectat pe magistrala de extensie poate prelua controlul magistralei.

$\overline{\text{BHE}}$: (A41) — este un semnal prin care microprocesorul 8086 stabilește modul de acces la datele de pe magistrala de date:

$\overline{\text{BHE}}$: A0	
0 : 0 —	acces pe 16 biți
0 : 1 —	D8: 15 conține octet date adresa impară
1 : 0 —	D0: 7 conține octet date adresa pară
1 : 1 —	neutilizat

$\overline{\text{INTA}}$: (A42) — este un semnal prin care se confirmă luarea în considerare a cererii de întrerupere mai prioritară de la sistemul de intreruperi. Este activ pe „0".

PCLK: (A45) — este un semnal de tact de ~2,5 MHz.

$\overline{\text{NMI}}$: (A48) — este un semnal prin care resursele modulului de bază fac cerere de întrerupere nemascabilă. Este activ pe „0" și reprezintă o stare pentru modulele de extensie master de pe magistrala sistemului.

3. Terminalele color utilizate pentru FELIX-PC

Spre deosebire de dispozitivele de afișare pe tub catodic (CRT) monocrome (de obicei alb/negru), tuburile color utilizează un ecran acoperit cu mai multe tipuri de fosfor. Principiul de funcționare a tuburilor color se bazează pe faptul că diferite tipuri de fosfor emit lumina colorată diferit. Combinația dintre lumina emisă de mai multe tipuri de fosfor poate genera o intreagă paletă de culori diferite.

În construcția tuburilor pentru televizoare se utilizează o mască perforată foarte fin pentru a obține culorile. Ecranul este acoperit cu triade foarte fine din fosfor de tipuri diferite. Astfel, un punct din triadă emite culoarea roșie (R), al doilea emite culoarea albastră (B). Cele trei puncte sînt dispuse foarte apropiate unele de altele, formînd o rețea de triade care generează punctele pe ecran (pixeli). Un tub color este echipat cu trei tunuri electronice, cîte unul pentru fiecare punct din triadă. Masca perforată este plasată în spatele ecranului acoperit cu triade de fosfor și are funcția de focalizare a celor trei tunuri electronice, astfel ca fiecare fascicol să fie focalizat numai pe punctul asociat din fiecare triadă. Controlînd intensitatea fiecărui fascicol se comandă intensitatea emisiei luminoase a fiecărui punct din triadă, obținîndu-se puncte de diferite culori pe ecran.

Dacă fiecare fascicol electronic este controlat binar (el poate fi activ sau inactiv, i.e. aprins sau stins) se obțin următoarele culori:

R G B	culoare	denumire în engleză
0 0 0	negru	black
0 0 1	albastru	blue
0 1 0	verde	green
0 1 1	turcoaz	cyan
1 0 0	roșu	red
1 0 1	purpuriu	magenta
1 1 0	cafeniu	brown
1 1 1	alb	white

După modul în care se transmit la tubul catodic informaţiile de aprindere/stingere a punctelor pe ecran se disting următoarele tipuri de dispozitive de afişare pe tub catodic:

— monitor RGB;
— monitor cu intrare video-compus;
— televizor color.

Monitor RGB. Informaţiile privind activarea celor trei tunuri electronice pentru fiecare triadă se transmit direct de la interfaţa cu monitorul. Pentru a putea funcţiona corect interfaţa trebuie să furnizeze monitorului semnalele de sincronizare. Sincronizarea se poate face prin semnale separate pe orizontală (HSYNC) şi pe verticală (VSYNC) sau printr-un semnal de sincronizare compus (CSYNC= =HSYNC+VSYNC). Deoarece semnalele pentru controlul intensităţii tunurilor electronice se transmit direct, la astfel de monitoare imaginea este de cea mai bună calitate.

Monitor cu video-compus. Aceste monitoare primesc informaţiile de sincronizare, luminozitate şi culoare codificate într-un semnal denumit video-compus. Acest semnal este decodificat de monitor pentru a obţine. separat, informaţiile necesare intern. Datorită codificării în interfaţa şi decodificării în monitor calitatea imaginii va fi scăzută faţă de monitoarele RGB. De asemenea, este necesar ca atît codificarea cît şi decodificarea să respecte acelaşi standard (PAL, SECAM sau NTSC cu variantele acestora), altfel monitorul va afişa doar imagini alb-negru.

Televizor color. Pentru afişarea color a imaginilor se poate utiliza un televizor color obişnuit dacă semnalul video-compus generat de interfaţă este utilizat pentru a modula o purtătoare de radio frecvenţă, cu frecvenţa corespunzătoare unuia din canalele VHF sau UHF disponibile la televizor. Televizorul va recepţiona prin antenă semnalul de radiofrecvenţă, va face demodularea pentru a obţine semnalul video-compus pe care îl va decodifica apoi pentru a obţine informaţiile de sincronizare, strălucire şi culoare necesare pentru a comanda cele trei fascicole electronice ale tubului. Prin operaţiile de codificare-modulare-demodulare-decodificare calitatea semnalului se deteriorează obţinînd astfel performanţe mai scăzute faţă de monitoare video-compus sau RGB. De asemenea, banda de trecere a amplificatorului video de la televizor este o limitare în ceea ce priveşte frecvenţa punctelor trimise de interfaţă.

La microcalculatorul FELIX-PC se poate conecta un monitor cu intrare video-compus (standardul PAL sau NTSC) sau un televizor color, utilizînd un modulator de radio frecvenţă extern pentru canalul dorit. De asemenea FELIX-PC poate utiliza un monitor RGB cu două trepte de intensitate (RGBI), obţinîndu-se în acest caz încă 8 culori, deci 16 culori în total. În acest caz cele 16 culori posibile sînt:

I R G B	culoare	denumire în egnleză
0 0 0 0	negru	black
0 0 0 1	albastru	blue
0 0 1 0	verde	green
0 0 1 1	turcoaz	cyan
0 1 0 0	roşu	red
0 1 0 1	purpuriu	magenta
0 1 1 0	cafeniu	brown
0 1 1 1	alb	white
1 0 0 0	gri	gray
1 0 0 1	bleu	light blue
1 0 1 0	verde deschis	light green
1 0 1 1	turcoaz deschis	light cyan
1 1 0 0	roşu aprins	light red
1 1 0 1	purpuriu deschis	light magenta
1 1 1 0	galben	yellow
1 1 1 1	alb strălucitor	light white

Pentru a genera cele 16 culori monitorul RGB trebuie să permită intrări pentru R, G, B, I şi SYNC.

Rezoluţia şi culorile la FELIX-PC. Interfaţa pentru terminalul grafic color constituie un modul de extensie standard în configuraţia microsistemului FELIX-PC. Această soluţie a fost aleasă pentru a asigura o mai mare flexibilitate în ceea ce priveşte subsistemul de afişare grafică. Adaptorul pentru monitor conţine 32 Kocteţi de memorie biport organizată ca „memory mapped display". Memoria de reîmprospătare a ecranului este inclusă în spaţiul de adresare al procesorului. Acest lucru permite o flexibilitate maximă în elaborarea programelor de manipulare a imaginilor pe ecran.

FELIX-PC poate să funcţioneze cu monitoare RGBI şi sincronizări (VSYNC, HSYNC) separate, cu monitoare color cu intrare video-compus sau televizoare color în sistemele NTSC (3,54 Mhz) sau PAL.

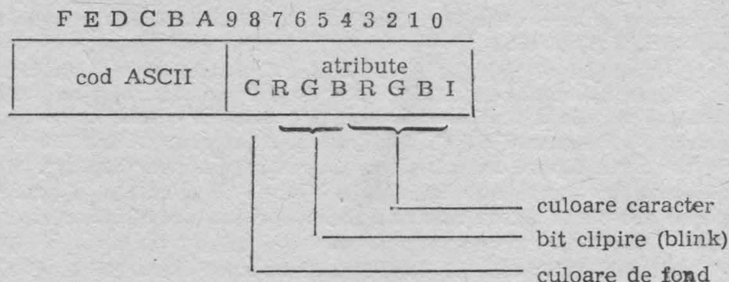
Pentru a fi posibilă utilizarea atît a unor monitoare performante cît şi a televizoarelor obişnuite este necesar ca rezoluţia cu care se afişează imaginea să aibă cel puţin două trepte. Modurile de lucru, rezoluţia şi culorilor sînt prezentate în următorul tabel:

mod	rezoluţie	format grafic	format text	culori
1	mică	—	40×25	8 — fond 16 — caracter 16 — chenar
2	mare	—	80×25	8 — fond 16 — caracter 16 — chenar
3	mică	320×200	40×25	16 — fond 3 — pt. punct (din 6 video) 3 — pt. punct (din 7 RGB)
4	medie	640×200	80×25	1 — pt. punct (din 16)
5*	mare	640×200	80×25	16 — fond 3 — pt. punct (din 6 video) 3 — pt. punct (din 7 RGB)
6*	medie	640×400	80×25	1 — pt. punct (din 16)

Modurile 5 şi 6 nu sînt implementate la versiunea actuală de interfaţă

Modurile 1 şi 2 permit lucrul în regim alfanumeric care de fapt este un regim semigrafic avînd în vedere că pe lîngă setul de caractere ASCII standard mai există o serie de caractere grafice care permit realizarea unor grafice de rezoluţie mică adecvate pentru multe aplicaţii. În aceste moduri memoria de reînprospătare a ecranului conţine codurile ASCII ale caracterelor iar generarea caracterelor se face printr-un generator hardware. Atributele de culoare sînt asociate caracterelor şi prin alegerea adecvată a culorilor de fond şi a culorilor caracterelor se pot obţine grafice sau texte multicolore.

În modul alfanumeric (semigrafic) fiecare caracter este reprezentat în memoria de reîmprospătare pe 2 octeţi. Primul actet conţine codul ASCII al caracterului iar al doilea octet conţine atributele de culoare şi efecte speciale.



Culoarea pentru chenar se selectează separat într-un registru separat destinat acestei funcții.

Toți biții din octetul de atribute comandă culori separate care apoi sînt combinate de circuitele video din interfață pentru a forma culorile de fond și de desenare.

Modurile 3—6 permit lucrul în mod grafic, dar desigur permit și afișarea de texte. Memoria de reînprospătare corespunzătoare a ecranului conține informația care se afișează direct pe ecran. Deci în acest caz caracterele alfanumerice (și semigrafice) sînt generate prin program și ele apar ca grafice în memoria de ecran.

Corespondența dintre punctele de pe ecran și informația din memoria de ecran este:

— 1 bit/punct (pixel) pentru modurile 4 și 6;

— 2 biți/punct (pixel) pentru modurile 3 și 5.

Pentru modurile 4 și 6 culorile de fond și chenar sînt întotdeauna negre iar culoarea cu care se desenează poate fi oricare din cele 16 culori posibile.

Pentru modurile 3 și 5 există 3 palete a cîte 3 culori pentru monitoarele RGB și 2 palete pentru monitoarele video și TV. Avînd în vedere că se poate selecta și una din cele 16 culori pentru fond, se pot obține desene multicolore formate practic din 4 culori simultan pe ecran. Codificarea culorilor pentru modurile 3 și 5 este prezentată în continuare:

B1 B0	culoare (paleta 0)	culoare (paleta 1)	culoare (paleta 2)*
0 0	culoare de fond	culoare de fond	culoare de fond
0 1	verde (green)	turcoaz (cyan)	turcoaz (cyan)
1 0	roșu (red)	purpuriu (magenta)	roșu (red)
1 1	cafeniu (brown)	alb (white)	alb (white)

Paleta 2 poate fi selectată numai pentru monitoarele RGB și nu este selectabilă prin BIOS. Pentru utilizarea acesteia utilizatorul trebuie să-și prevadă rutinele corespunzătoare.

Pentru modurile alfanumerice (1 și 2), în memoria asociată terminalului grafic se opt organiza 8, respectiv 4 pagini ecran. Dintre acestea este activă (afișată pe ecran) una singură la un moment dat. Selectarea paginii de lucru se face prin programare.

4. Sistemul de programe de intrare/ieșire (I/E) de bază (BIOS)

BIOS-ul este memorat în memoria permanentă a sistemului FELIX-PC și conține:

— un program de test al resurselor din configurația standard controlate de către BIOS;

— un încărcător pentru sistemul de operare rezident pe disc;
— procedurile de tratare a întreruperilor:
— nivel 0 — ceasul de timp real;
— nivel 1 — tastatura;
— nivel 6 — disc flexibil.
— procedurile pentru comanda și controlul execuției operațiilor de intrare/ieșire pentru echipamentele periferice conectate la sistem.

Lansarea programului de test se face automat la pornirea sistemului. În cadrul acestui program se realizează:
— testarea sumară a microprocesorului;
— testarea și inițializarea canalului DMA și a contorului programabil în vederea execuției operației de reîmprospătare a memoriei RAM dinamice;
— testarea memoriei RAM;
— testarea sistemului de întreruperi și inițializarea tabelei care conține adresele rutinelor pentru tratarea întreruperilor;
— testarea adaptorului pentru terminalul grafic color;
— testarea sumară a interfeței pentru disc flexibil.

În cazul în care se detectează erori pe parcursul programului de test, în funcție de gravitatea erorii se semnalează eroarea și se continuă testul sau se semnalează eroarea și se trece în starea HALT. Toate erorile detectate înainte de testarea interfeței pentru terminalul grafic sînt semnalate sonor.

Dacă execuția programului de test se încheie fără erori se va apela programul încărcător pentru sistemul de operare rezident pe disc.

Cele 8 nivele de întrerupere tratate de către sistemul de întreruperi corespund nivelelor 8—15 ale microprocesorului, astfel că procedura pentru tratarea intreruperii pentru ceasul de timp real poate să fie apelată și cu ajutorul instrucțiunii INT 8, procedura pentru tratarea întreruperii de la tastatură poate să fie apelată și cu ajutorul instrucțiunii INT 9 iar procedura pentru tratarea întreruperii de la discul flexibil poate să fie apelată cu instrucțiunea INT 0EH.

Accesul din programe exterioare BIOS-ului la procedurile conținute în BIOS se face prin intreruperi software (instrucțiuni INT). Principalele puncte de intrare în BIOS sînt:

INT 10H — intrare pentru proceduri terminal grafic color;
INT 11H — intrare pentru determinarea configurației sistemului;
INT 12H — intrare pentru determinarea capacității memoriei;
INT 13H — intrare pentru proceduri disc;
INT 14H — intrare pentru interfața serială;
INT 16H — intrare pentru tastatură;
INT 17H — intrare pentru imprimantă;
INT 19H — intrare pentru incărcătorul sistemului de operare rezident pe disc;
INT 1AH — intrare pentru citire/inițializare oră curentă (ceas de timp real).

Tabela care conține adresele rutinelor de tratare a întreruperilor este alcătuită din intrări formate din cîte 4 octeți (adresa relativă în segment, adresa de segment) și este memorată în memoria RAM începînd de la adresa 0. În cadrul acestei tabele în afară de adresele rutinelor de tratare a întreruperilor și a punctelor de intrare în BIOS prezentate anterior mai sînt memorate:
— adresa procedurii pentru tratarea întreruperii nemascabile (INT 2);
— adresa unei proceduri care realizează copierea la imprimantă a caracterelor afișate pe ecran (INT 5);
— adresa tabelei care conține parametri pentru programarea interfeței pentru terminalul grafic color (INT, 1DH);
— adresa tabelei care contne parametri pentru programarea interfeței de disc flexibil (INT 1EH).

Pentru mărirea disponibilității sistemului sînt în lucru următoarele module de extensie:
— interfață pentru imprimanta paralelă;
— interfață pentru I/E analogice și numerice;
— interfață specializată pentru aplicații medicale.

5. Sistemul de programe de bază și aplicații

Sistemul de programe implementat pe FELIX-PC are la baza sistemele de operare PC-DOS și MS-DOS și include:

— utilitarele sistemului de operare pentru interfața cu operatorul, gestiunea și intreținerea fișierelor, programe de test etc.

— facilități de execuție și depanare a programelor;

— translatoare pentru programe în limbaj de asamblare și pentru limbajul BASIC;

— interpretor BASIC cu facilități pentru prelucrări grafice;

— mediu pentru dezvoltarea programelor în PASCAL;

— mediu pentru dezvoltarea programelor în EDISON;

— programe de aplicații pentru:

— editarea și prelucrarea textelor;

— baze de date;

— culegere și validarea datelor;

— prelucrări grafice;

— aplicații economice.

6. Concluzii

Sistemul FELIX PC este funcțional sub forma de model de laborator cu resursele hardware și software prezentate anterior și este în curs de asimilare în fabricație la ICE.

Datorită soluțiilor tehnologice ce vizează implementarea sistemului este de așteptat ca fiabilitatea sistemului să fie ridicată, constituind o alternativă pentru diverse aplicații industriale.

Compatibilitaea cu microsisteme similare cu o largă răspîndire cum ar fi: IBM-PC, SANYO-550, OLIVETTI M24, CORONA, etc. oferă o mare disponibilitate software sistemului.

FELIX-PC este, pe de o parte, un nou tip de microcalculator (calculator personal-profesional) cu performanțe ridicate, după cum reiese din text, dar pe de altă parte el reprezintă un simbol al entuziasmului și profesionalismului cu care mulți cercetători din domeniul tehnicii de calcul înțeleg să-și facă meseria. Obiectivul acestei lucrări ne permite ca pe lîngă caracteristicile hardware și software ale lui FELIX-PC să enumerăm și alte aspecte „hartware" sau „humanware" în legătură cu proiectarea, introducerea în fabricație și utilizarea lui FELIX-PC. Pe lîngă colectivul care de la Inst. Politehnic, care l-a conceput, se impune să-i amintim (deși ei merită mai mult) pe *Alexandru Anghel, Gabi Draghicescu, Titi Alupului* și *Anca Negulescu* de la ICE care au avut o contribuție importantă la finalizarea lui FELIX-PC.

Iar, fără sprijinul nemijlocit al conducerii ICE, reprezentată prin *Florea Tănase, Dorin Mihu* și *Tudorel Domocoș*, *FELIX-PC* ar fi rămas „probabil" pe hîrtie.

Bibliografie

[1] A. Petrescu și colectiv, Microcalculatorul FELIX-PC, *Lucrările CNETAC*, Noiembrie 1984.
[2] A. Petrescu și colectiv, Structura și arhitectura sistemului FELIX-PC, Lucrările SAII, Mai 1985.
[3] G. Williams. A Closer Look tat the IBM Personal Computer, *BYTE*, January 1982.
[4] * * * — Guide to operations, *IMB Corporation.*
[5] J. E. Pettersen, The IBM Personal Computer, Evaluation and observations, *Tenth EUROMICRO Symposium on microprocessing and microprogramming.* 27—30 August 1984.
[6] W. Richardson, Future Computing, Market Research: *Personal Computers.*
[7] * * * — PC-DOS Reference Manual. *IBM Corporation.*
[8] * * * — The IBM Compatible Universe, *PC Magazine,* Vol. 3, 1984.

MICROCALCULATORUL PERSONAL HC-80
(Familia aMIC, Felix Student şi HC-80)

prof. dr. ing. Adrian Petrescu
as. ing. Francisc Iacob
I.P.B.

1. Introducere

În cadrul Catedrei de Calculatoare din Institutul Politehnic Bucureşti au fost proiectate şi realizate o serie de microcalculatoare de tip personal, de cost scăzut, pentru a fi utilizate, în primul rînd, în instituţiile de învăţămînt superior şi mediu, dar şi în laboratoare, instituţii de cercetare şi proiectare, întreprinderi sau chiar în producţie pentru controlul unor procese industriale.

Aceste microcalculatoare sînt aMIC, FELIX-STUDENT şi HC-80. Din punct de vedere al arhitecturii interne sînt identice, prezentînd caracteristici ca:

— microprocesor Z80;
— memorie EPROM+RAM de 64 kocteţi;
— utilizează ca dispozitiv de afişare un televizor alb/negru sau color (în cazul microcalculatorului HC-80);
— ca dispozitiv de memorie externă se foloseşte un casetofon audio obişnuit;
— tastatură simplă;
— interfaţă serială;
— interfaţă paralelă;
— difuzor pentru aplicaţii acustice;
— software de bază şi de aplicaţii compatibil.

Structura acestor sisteme este modulară, blocurile componente fiind interconectate prin intermediul unei magistrale unice. Modificări în cadrul sistemului se pot efectua uşor, prin adăugarea sau eliminarea unor module.

Unele dintre aceste microcalculatoare pot fi utilizate ca sisteme pe o singură placă, dedicate, pentru care programele de aplicaţii sînt scrise în limbaj de asamblare, ceea ce asigură o viteză mare de execuţie. Altele reprezintă maşini BASIC, dar permit încărcarea şi rularea şi altor tipuri de programe.

Microcalculatorul aMIC se află în prezent în producţie de serie la Fabrica de Memorii Timişoara, iar FELIX-STUDENT are perspective de a fi produs în cadrul Atelierului de producţie din Facultatea de Automatică.*

Microcalculatorul HC-80 se află în prezent în faza de punere în fabricaţie de serie la Întreprinderea de Calculatoare Electronice Bucureşti.

2. Descriere HARDWARE

Microcalculatorul HC-80 se prezintă sub forma unui sistem pe o singură plachetă pe care se află unitatea centrală, memoria şi interfeţele şi la care se conectează următoarele echipamente periferice:

— tastatură pentru introducrea de date şi comenzi;

* Cartea în două volume, „Totul despre... calculatorul personal aMIC" de A. Petrescu şi colectiv apare în tr. IV. '85 la Editura Tehnică.

— televizor sau monitor color sau alb/negru (în acest caz se afişează nuanţe de gri);
— casetofon pentru salvarea de programe şi refacerea ulterioară a acestora;
— modem pentru transmiterea/recepţionarea datelor prin linie telefonică;
— difuzor pentru diverse aplicaţii acustice.

Modulele componente ale sistemului sînt:

— unitatea centrală de prelucrare;
— memoria EPROM;
— memoria RAM;
— logica de afişare la televizor;
— interfaţa pentru tastatură;
— interfaţa pentru casetofonul audio;
— interfaţa pentru difuzor.

Toate aceste module sînt interconectate printr-o magistrală care cuprinde:

— liniile de adresă;
— liniile de date;
— liniile de comenzi;
— liniile de alimentare.

Magistrala sistemului este scoasă în exterior la un conector, ceea ce permite interfaţarea de echipamente la acest microcalculator.

2.1. Unitatea centrală de prelucrare

Unitatea centrală de prelucrare este construită în jurul unui microprocesor Z80A, utilizat la frecvenţa de 3,5 MHz.

Semnalele disponibile la pinii microprocesorului utilizate în cadrul microcalculatorului sînt următoarele:

— A0—A15 reprezintă magistrala de adrese utilizată pentru adresarea la memorie sau la un port de intrare/ieşire. Această magistrală poate adresa direct 64 kocteţi de memorie sau 256 de porturi de intrare/ieşire. De asemenea, liniile de adrese A0—A6 sînt utilizate pentru reîmprospătarea memoriei dinamice.

— D0—D7 reprezintă magistrala de date bidirecţională, utilizată pentru transferul de date între microprocesor şi memorie sau dispozitive periferice.

— $\overline{M1}$ împreună cu $\overline{MREQ}$ activ indică un ciclu de fetch, iar $\overline{M1}$ împreună cu $\overline{IORQ}$ activ indică un ciclu de luare în considerare a unei întreruperi.

— $\overline{MREQ}$ indică un ciclu de acces la memoria EPROM sau RAM.

— $\overline{IORQ}$ indică un ciclu de intrare/ieşire. De notat faptul că semnalele $\overline{MREQ}$ şi $\overline{IORQ}$ nu sînt active niciodată în acelaşi timp.

— $\overline{RD}$ indică faptul că unitatea centrală de prelucrare citeşte date dintr-o locaţie de memorie sau dintr-un port de intrare/ieşire.

— $\overline{WR}$ indică faptul că microprocesorul se află într-un ciclu de scriere într-o locaţie de memorie sau într-un port de intrare/ieşire.

— $\overline{RFSH}$ este utilizat pentru reîmprospătarea informaţiei în blocul de memorie suplimentară.

— $\overline{INT}$ reprezintă pinul pentru cererea de întrerupere mascabilă prin program. Microprocesorul are setat modul 1 de răspuns la întreruperi, iar în rutina de tratare este scanată tastatura pentru identificarea unei taste apăsate.

— $\overline{RESET}$ este controlat de către un comutator de pe carcasa calculatorului şi activarea acestui semnal are ca efect relansarea microprocesorului de la adresa 0000H.

— CLK reprezintă intrarea de ceas a microprocesorului, prin acest pin furnizîndu-se de către logica de afişare la televizor un semnal cu frecvenţa de 3,5 MHz.

Celelalte semnale disponibile la pinii microprocesorului nu sînt utilizate direct în funcţionarea microcalculatorului, dar acestea sînt furnizate în exterior printr-un conector. Semnalele furnizate la conectorul extern sînt:

A11, A9, $\overline{BUSACK}$, nc, A4, A5, A6, A7, $\overline{RESET}$, $\overline{BUSREQ}$, nc, nc, nc, nc, GND, nc, A3, A2, A1, A0, CLK, GND, GND, nc, nc, +5V, A12, A14,

nc, A10, A8, $\overline{RFSH}$, $\overline{M1}$, +12V, +12V, $\overline{WAIT}$, —5V, $\overline{WR}$, $\overline{RD}$, $\overline{IORQ}$, $\overline{MREQ}$, $\overline{HALT}$, NMI, INT, D4, D3, D5, D6, D2, D1, D0, nc, nc, D7, A13, A15,

Pinii conectorului notaţi cu nc (neconectat) sînt prevăzuţi pentru extinderi ulterioare ale microcalculatorului.

2.2. Memoria EPROM

Modulul de memorie EPROM are capacitatea de 16 kocteţi şi este construit cu opt circuite 2716 (fiecare avînd capacitatea de 2 kocteţi). Acest modul se situează în spaţiul de adresare al microprocesorului între adresele 0000H și 3FFFH.

Caracteristicile principale ale acestor circuite sînt:

— capsulă standard cu 24 pini;
— capacitatea 16384×1 biţi;
— timp de acces 450 ns;
— putere maximă disipată 500 mW;
— intrările/ieșire compatibile TTL;
— ieșirile sînt cu 3 stări;
— o singură tensiune de alimentare +5V.

Decodificarea biţilor superiori de adresă A11-A15 se face printr-un circuit 74LS138, care furnizează semnalele de selecţie pentru cipurile de memorie.

Acest modul de memorie conţine un interpretor BASIC performant, care include și rutinele de intrare/ieșire pentru televizor, tastatură, casetofon precum și generatorul de caractere.

2.3. Memoria RAM

Memoria RAM ocupă spaţiul de adresare cuprins între adresele 4000H și FFFFH. Fizic este format din două blocuri:

— blocul de memorie de bază, conţinînd un singur modul de 16 kocteţi situat între adresele 4000H și 7FFFH;

— blocul de memorie suplimentară, format din două module de cîte 16 kocteți fiecare, între adresele 8000H și BFFFH, respectiv C000H și FFFFH.

Întreaga memorie RAM este construită cu circuite dinamice 4116, dar se poate utiliza ca variantă de implementare pentru blocul de memorie suplimentară un set de opt circuite 4164.

Caracteristicile cipurilor 4186 sînt:

— capacitatea 16384*1 biţi;
— capsula standard 16 pini;
— patru pini de alimentare: +5V, +12V, —5V și GND;
— timp de acces: 150/200/250 ns funcţie de tipul circuitului;
— ciclul de memorie: 320/375/410 ns;
— consum scăzut de energie, 462 mW (activ) / 20 mW (inactiv);
— necesită 128 cicluri de reîmprospătare la interval de 2 ms.

Blocul de memorie de bază conţine o zonă rezervată pentru memoria ecran, variabilele utilizate de interpretorul de BASIC și o zonă disponibilă pentru programele utilizator. Prezenţa acestui bloc este obligatorie pentru funcţionarea sistemului.

Reîmprospătarea informaţiei din acest bloc este realizată de către logica de afișare. Ciclul de memorie este împărţit în două subcicluri:

— citirea informaţiilor necesare pentru afișarea pe ecranul televizorului;

— accesul efectuat de către microprocesor pentru citirea sau scrierea din/în memorie.

Dacă accesul din partea microprocesorului are loc într-un moment de timp cînd acest lucru nu este permis (se citesc date pentru afișare), activitatea unităţii centrale de prelucrare se suspendă, prin oprirea ceasului de la pinul CLK. Ceasul este repornit în momentul în care se ajunge în subciclul de acces și transferul are loc în continuare pentru citirea sau scrierea unui octet.

Al doilea bloc de memorie conţine o zonă de 32 kocteţi la care accesul din partea microprocesorului are loc fără trecerea în starea de aşteptare. Reîmprospătarea informaţiei este realizată de către microprocesor pin semnalul scos pe pinul RFSH şi biţii de adresă A0-A6.

2.4. Logica de afişare la televizor

Microcalculatorul HC-80 are posibilitatea să afişeze informaţiile la un televizor color. În acest scop în blocul de memorie de bază există o zonă de memorie ecran, în care fiecărui bit îi corespunde un punct din imagine. Utilizatorul are acces la nivel de bit, deci poate programa oricare din aceşti biţi să fie 0 sau 1, ceea ce corespunde la puncte stinse sau aprinse pe ecran. În acest fel se pot trasa diferite desene prin program. Rezoluţia grafică a imaginii este de 192 de linii a 256 de puncte, ceea ce înseamnă o capacitate a memoriei ecran de 6 kocteţi.

Memoria ecran este situată între adresele 4000H şi 57FFH. În afară de această zonă, logica de afişare la televizor utilizează şi o aşa numită zonă de atribute, în care pentru fiecare matrice de 8×8 puncte se definesc o serie de caracteristici:

— culoarea fondului;

— culoarea caracterului;

— clipire;

Aceasta zonă de atribute conţine 768 de octeţi şi se întinde între adresele 5800H şi 5AFFH. Astfel se pot selecta pentru fiecare matrice elementară de 8×8 puncte două culori din totalul de opt culori disponibile.

Zona de ecran este împărţită în trei subzone de cîte opt rînduri de caractere. Fiecare rînd de caractere este afişat prin opt linii de televiziune.

În regim alfanumeric se pot afişa 24 de rînduri a cîte 32 de caractere. Setul de caractere cuprinde literele mari, literele mici, cifrele, semnele speciale. Caracterele pot fi redefinite de către utilizator.

2.5. Interfaţa cu tastatura

Microcalculatorul HC-80 utilizază ca dispozitiv pentru introducerea de date o tastatură simplă, formată dintr-o matrice de 5×8 taste. Liniile matricii sînt conectate la liniile de adresă A8-A15, iar coloanele pot fi citite pe magistrala de date în liniile D0—D4.

La fiecare 20 ms logica de afişare la televizor furnizează o întrerupere către microprocesorul Z80A pe pinul INT. Anterior sistemul de întreruperi a fost setat la modul 1 de răspuns. La sosirea unei cereri de întrerupere microprocesorul execută rutina de tratare a întreruperii de la adresa 38H pin care se determină dacă există cel puţin o tastă apăsată. În cazul în care există o tastă, apăsată se determină codul tastei prin căutare într-o tabelă de coduri.

Fiecare tastă are mai multe funcţii, iar dacă se lucrează sub interpretorul de BASIC apăsarea unei singure taste permite introducerea unui cuvînt cheie, ceea ce conferă o mai mare viteză de introducere a programelor.

2.6. Interfaţa cu casetofonul audio

Interfaţa cu casetofonul audio permite salvarea de programe şi refacerea acestora cu o viteză de aproximativ 1500 Bauds şi cu o fiabilitate foarte bună.

3. Software-ul de sistem al microcalculatorului personal HC-80

3.1. Generalităţi

Pentru utilizator microcalculatorul HC-80 se comportă ca o „maşina“ BASIC, avînd interpretorul pentru acest limbaj stocat în memoria cu conţinut permanent-EPROM.

Sistemul este prevăzut cu un editor extrem de performant, orientat puternic către utilizator, ceea ce ușurează în cea mai mare măsură punerea la punct a programelor, prin semnalarea erorilor sintactice la nivelul fiecărei linii.

Tastatura microcalculatorului are un caracter funcțional, fiind orientată pe instrucțiunile limbajului BASIC. După o perioadă de acomodare, utilizatorul va putea constata cu ușurință eficiența acesttei soluții. În esență tastatura este organizată conform standardului QWERTY, dispunînd însă numai de 40 de caractere alfa-numerice. Fiecare tastă poate avea pînă la cinci semnificații diferite, în funcție de modul în care se află sistemul, mod semnalat de caracterul specificat de cursor.

În afara limbajului BASIC, pe microcalculatorul HC-80 mai sînt implementate o serie de limbaje de nivel înalt: Pascal, FORTH, micro-PROLOG etc. De asemenea, pentru programarea în limbaj de asamblare sînt implementate un asamblor și un editor. Asamblorul operează cu mnemonicele setului de instrucțiuni ale microprocesorului Z-80.

3.2. Limbajul BASIC

O linie a unui program BASIC poate conține pînă la 704 caractere alfanumerice Numărul liniilor unui program poate fi cuprins între 1—9999.

BASIC HC-80 manipulează numere reprezentate în gama 10 ↑ 38-4*10 ↑ (—38), cu o precizie de 9 pînă la 10 cifre zecimale. Numerele sînt stocate în memorie în formatul cu virgulă mobilă, alocîndu-se un octet pentru exponent și patru octeți pentru mantisă.

Variabilele numerice au nume de lungime arbitrară, începînd cu o literă și continuînd cu litere și cifre. Comenzile de spațiu și culori sînt ignorate, toate literele fiind convertite în litere mici.

Variabilele de comandă pentru ciclurile FORT-NEXT au nume cu lungimea de un caracter.

Tablourile numerice au nume formate dintr-un singur caracter. Ele pot avea un număr nelimitat de dimensiuni, de mărimi arbitrare.

Șirurile pot avea o lungime nelimitată și ca nume o literă urmată de semnul $. Într-un tablou dat toate șirurile au aceeași lungime fixă, care este specificată într-o dimensiune finală, suplimentară, a instrucțiunii.

Funcții

ABS	ACS	AND	ASN	ATN	ATTR	BIN
CHR$	CODE	COS	EXP	FN	IN	INKEY$
INT	LEN	LN	NOT	OR	PEEK	PI
POINT	RND	SCREEN	SGN	SIN	SQR	STR$
TAN	USR	VAL	VAL$	—		

Operații binare.

+ — × / ↑ =

= > < <= >= <>

Prioritățile funcțiilor și ale operațiilor

Operația	Prioritatea
Indexare și separare	12
Toate funcțiile cu excepția lui NOT și minus unar	11
↑	10
Minus unar (folosit pentru negație)	9
×, /	8

+, — (minus folosit pentru a scădea un număr din altul)	6
=, >, <, <=, >=, <>	5
NOT	4
AND	3
OR	2

INSTRUCŢIUNI

În cele ce urmează se vor folosi următoarele notaţii:

a — litera singulară,
v — variabilă
x, y, z — expresie numerică.
m, n — expresie numerică rotunjită cel mai apropiat întreg
e — expresie,
f — expresie de tip şir evaluat,
s — secvenţă de instrucţiuni separate prin două puncte (:),
c — secvenţă de obiecte culori, separate prin virgulă (,) sau punct şi virgulă (;). Un obiect culoare are forma unei instrucţiuni: PAPER, INK, FLASH, BRIGHT, INVERSE sau OVER.

BEEP x, y	BORDER m	BRIGHT
CAT	CIRCLE x, y, z	CLEAR
CLEAR n	CLOSE	CLS
CONTINUE	COPY	DATA e1, e2, e3...
DEF FN a (a1,... ak)=e	DELETE f	DIM a (n1, ... nk)
INVERSE n	LET v=e	LIST
LIST n	LLIST	LLIST n
LOAD f	LOAD f DATA ()	LOAD f DATA $ ()
LOAD f CODE m, n,	LOAD f CODE m	LOAD f CODE
LOAD f SCREEN$	LPRINT	MERGE f
MOVE f1, f2	NEW	NEXT a
OPEN	OUT m, n	OVER n
PAPER n	PAUSE n	PLOT c, m, n
POKE m, n	PRINT...	RANDOMIZE
RANDOMIZE n	READ v1, v2, ... vk	REM...
RESTORE	RESTORE n	RETURN
RUN	RUN n	SAVE f
SAVE f LINE m	SAVE f DATA ()	SAVE f DATA $ ()
SAVE f CODE m, n	SAVE f SCREEN$	STOP
VERIFY		

Interpretorul BASIC HC-80 furnizează 29 mesaje de eroare în clar. Limbajul BASIC este folosit pentru scrierea unor pachete de programe de aplicaţie extrem de puternice în tehnică, ştiinţă, gestiune economică, învăţămînt, medicină, agricultură etc.

4. Concluzii

Microcalculatorul personal HC-80 va intra în fabricaţie de serie, încă în acest an, la Întreprinderea de calculatoare electronice din Bucureşti. Prin biblioteca de programe de sistem şi aplicaţii de care dispune şi prin performanţele sale, inclusiv extensiile hardware, HC-80 se situează la nivelul celor mai reuşite calculatoare personale din aceasta categorie, realizate pe plan mondial. Limbajele Pascal, FORTH şi micro-PROLOG, pentru HC-80, vor fi descrise, pe larg, într-o lucrare viitoare.

Ciclul
„REȚELE DE CALCULATOARE"

MEMENTO DE TELEPRELUCRARE
(Partea a II-a)*

A Davidoviciu, A. Borozan, O. Borozan, R. Ciocea, M. Constantinescu, M. Ivan, A. Lazăr, Șt. Panait, D. Trifănescu
I.T.C.I.

4. TERMINALE

4.1. GENERALITĂȚI

Un terminal este un dispozitiv de afișare alfanumeric, grafic sau semigrafic pentru teletransmisie date, care permite introducerea, transmiterea și recepționarea informației în scopul utilizării eficiente a băncilor de date, fișierelor și altor informații stocate în memoria calculatorului.

Cuplarea unui terminal la calculator se realizează fie prin conectare directă, fie la distanță prin intermediul modemurilor.

Terminalele tratate în continuare vor fi simple sau paginate [tip display (pozitionare x,y) (Ex: DAF 2010, DAF 2015, DAF 2020, VDT 40), tip teletype (fără poziționare x,y) (Ex: CENTRONICS 761)]

Schema bloc a unui terminal tipic este prezentată în figura 4.1. (DAF 2010). [30]

Afișarea se poate face pe un ecran (tub catodic) la terminalele de tip display, pe hîrtie la terminalele de tip teletype sau combinate.

Tastatura unui terminal cuprinde următoarele tipuri de taste:

- alfanumerice (26 litere din alfabet);
- numerice (10 cifre);
- speciale: semne de punctuație, operatori aritmetici;
- funcționale.

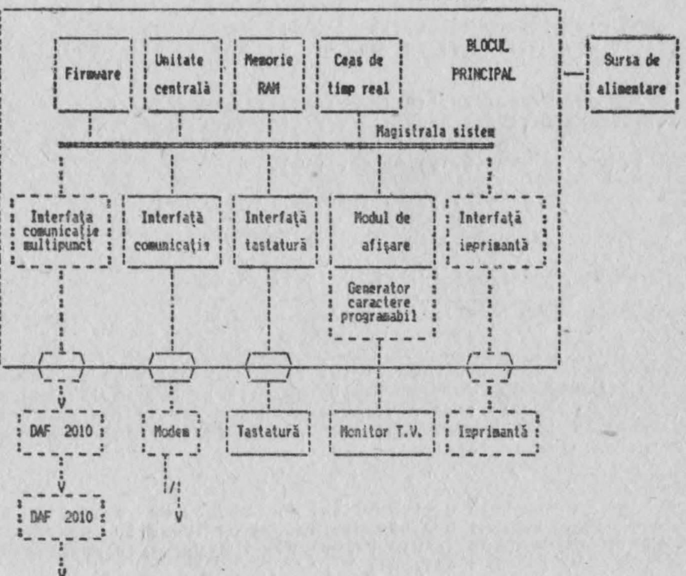


Figura 4.1 Structură terminal DAF 2010

* Partea I și cuprinsul complet în AMC 48, pag. 243—306.

TAB determină mutarea cursorului la poziția următoare de tabulare orizontală.

DEL (RUBOUT) șterge ultimul caracter introdus.

LF (LINE FEED) poziționare cursor pe linie nouă pe aceeași coloana (x, y) -> (x, y+1).

CR (RETURN) poziționare cursor la început de rînd (x, y) -> (1, y).

HOME poziționare cursor la început ecran (coloana 1 și linia 1 în mod paginat și coloana 1 și ultima linie în mod defilare).

SPACE caracterul spațiu (deobicei tasta SPACE este o tastă neinscripționată și mai mare decît restul tastelor).

↑ cursor în sus (x, y) -> (x, y-1).

↓ cursor în jos (x, y) -> (x, y+1).

-> cursor la dreapta (x, y) -> (x+1, y).

<- cursor la stînga (x, y) -> (x-1, y).

BS (BACK SPACE) determină mutarea cursorului cu o poziție la stînga.

BREAK determină forțarea liniei de transmisie date în stare zero logic pentru o durată egală cu circa 300 milisecunde (detectabilă hardware fără a se genera un cod specific). Funcția BREAK este folosită deobicei pentru a forța întreruperea recepției datelor la terminal.

Pentru terminalele de tip display, modul de execuție á funcțiilor terminalului depinde de regimul de lucru în care se află. Pentru descrierea regimurilor de lucru, este necesară precizarea următoarelor noțiuni:

- liniile de pe ecran sînt numerotate de sus în jos de la 1 la n (numărul maxim de linii);
- coloanele sînt numerotate de la stînga la dreapta de la 1 la m (numărul maxim de coloane);
- poziția curentă este poziția în care se află cursorul;
- poziția următoare este poziția pe care se va deplasa cursorul după afișarea unui caracter pe poziția curentă
- început ecran este poziția pe linia 1 coloana 1;
- sfîrșit ecran este poziția de pe linia n coloana m.

În continuare se descriu tastele funcționale uzuale (care se regăsesc la majoritatea terminalelor).

Codurile caracterelor afișabile se obțin prin acționarea tastelor alfanumerice. Tasta specială **SHIFT** ajută, prin apăsarea ei simultană cu o altă tastă, la selecția literelor mari/mici și a caracterelor de pe tastele cu dublă inscripționare (pentru fiecare terminal în parte se va explica efectul tastei SHIFT, acesta fiind diferit de la terminal la terminal).

Tastele **SHIFT LOCK** selectează emisia unuia din cele două caractere de pe tastele cu dublă inscripționare iar tasta CAPS a literelor mari/mici. De obicei tasta CAPS este înzestrată cu un indicator luminos, care arată ultima acționare a tastei. Pentru fiecare terminal în parte se va arăta modul de folosire a tastelor CAPS și SHIFT LOCK.

În general, tastele terminalelor sînt de tip AUTOREPEAT (la apăsarea prelungită a unei taste, caracterul corespunzător va fi repetat cu o viteza mai mică de 20 caractere pe secundă). Alte echipamente au o tastă specială **REP (REPEAT)** care prin apăsarea simultană cu o altă tastă produce repetarea caracterului corespunzător tastei respective (pentru fiecare terminal se va specifica modul de realizare a regimului de repetare caractere).

Prin apăsarea simultană a tastei speciale **CTRL** și a unei taste care generează cod corespunzător coloanelor 4 și 5 din tabelul de coduri ASCII (Tabelul 6.2.) se obțin codurile de comandă de teletransmisie (coloanele 0 și 1 din tabelul de coduri ASCII). Caracterele generate prin folosirea tastei CTRL sînt:

CTRL @	NUL	CTRL P	DLE
CRTL A	SOH	CTRL Q	DC1
CTRL B	STX	CTRL R	DC2
CRTL C	ETX	CTRL S	DC3
CTRL D	EOT	CTRL T	DC4
CTRL E	ENQ	CTRL U	NAK
CTRL F	ACK	CTRL V	SYN
CTRL G	BEL	CTRL W	ETB
CTRL H	BS	CTRL X	CAN
CTRL I	HT	CTRL Y	EM
CTRL J	LF	CTRL Z	SUB
CTRL K	VT	CTRL [	ESC
CTRL L	FF	CTRL \	FS
CTRL M	CR	CTRL]	GS
CTRL N	SO	CTRL ^	RS
CTRL O	SI	CTRL _	US

ESC (ESCAPE) este un caracter de control care, de obicei, precede un alt caracter pentru ca împreună să determine o funcție specială.

Regimuri de lucru

DEFILARE/PAGINAT

Se referă la modul de lucru cu informaţia pe ecran. În regim DEFILARE se simulează funcţionarea maşinii de scris (cînd poziţia curentă este pe ultima linie de ecran şi se comandă trecerea pe următoarea linie atunci se produce deplasarea în sus cu o linie a întregului text de pe ecran, cu pierderea primei linii).
În regim PAGINAT, cînd poziţia curentă este pe ultima linie de ecran, linia următoare este linie de început ecran.

AUTO LINE FEED

În acest regim, după recepţia unui cod CR se execută automat şi un LF. Este valabil pentru toate tipurile de terminale.

ON/OFF LINE

În regim de lucru ON LINE terminalul, cuplat la calculator, este pregătit pentru a realiza transferul de informaţie cu acesta. Cînd terminalul nu este cuplat la calculator, el se afla în regim OFF LINE. Este valabil pentru toate tipurile de terminale.

CU/FARA ECOU

În regim de lucru CU ECOU (duplex în unele documentaţii), la apăsarea unei taste, codul corespunzător este transmis în linia de comunicaţie iar pe ecran/hîrtie va fi afişat după retransmiterea lui de către calculator (nu întotdeauna calculatorul retransmite în ecou caracterul pe care l-a recepţionat). In regim FARA ECOU(semi-duplex în unele documentaţii) caracterul emis pe linie se afişează pe ecran/hîrtie în momentul tastării. În ambele cazuri caracterul recepţionat de la calculator se afişeaza pe ecran/hîrtie.

4.2. CENTRONICS 761

CENTRONICS 761 este un terminal de tip teletype. [20]

4.2.1. Caracteristici

. Mesajele introduse de la tastatură cît şi mesajele trimise de la calculator se afişeaza pe hîrtie pe linii de maxim 132 caractere.
. Interfaţa de comunicaţie cu calculatorul este standard CCITT V.24 / EIA RS 232C
. Regim de lucru:
  - cu ecou;
  - fără ecou (protocol X-ON/X-OFF).
. Tastatura alfanumerică cu 79 taste, dintre care 64 sînt caracterele alfanumerice din setul ASCII
. Are tastă REPT (de repetare)
. Formatul de tipărire este de 10 caractere/inch şi 6 linii/inch, cu o viteză de 60 caractere pe secundă
. Combinarea tastelor ALL CAPS şi SHIFT determină afişarea caracterelor în modul următor:

CAPS	SHIFT	Litere	Alte caractere
NU	NU	mici	caracterele de jos
NU	DA	mari	caracterele de sus
DA	NU	mari	caracterele de jos
DA	DA	mari	caracterele de sus

Tasta CAPS LOCK are un indicator luminos care arată cînd tasta este apăsata.

4.2.2. Comutatoare. Programare

Pe tastatură (în partea stîngă) se află 6 taste speciale şi 6 indicatoare luminoase.

Tastele speciale folosesc pentru stabilirea regimului de lucru al terminalului:

- **SELECT** selectare terminal (operatorul îl anunță operațional);
- **ON LINE** stabilire legatură cu calculatorul;
- **ALT CHAR** selectare set caractere alternat (dacă opțiunea hardware este prezentă);
- **OVER RIDE** poziționare cap de scriere la capătul din dreapta (pentru a putea vedea uşor textul deja imprimat);
- **TOF** poziționare la început de pagină;
- **LINE FEED** linie nouă în regim OFF LINE.

Indicatoarele luminoase au următoarea semnificație:

- **POWER** terminalul este sub tensiune (comutatorul de pornire terminal este în spatele terminalului în partea stîngă);
- **ALERT** terminalul este fără hîrtie;
- **READY** terminalul este în legătură fizică cu calculatorul;
- **SELECT** terminal pregătit de lucru;
- **ON LINE** terminal este pregătit pentru a lucra cu linia de comunicație;
- **ALT CHAR** s-a selectat setul de caractere alternat.

Între blocul de taste speciale şi blocul de taste alfanumerice se află două blocuri de microcomutatoare, unul după altul, numerotate de sus în jos 1...8 pentru blocul de sus (A) şi 1...8 pentru blocul de jos (B). Un comutator apăsat spre stînga este OFF (deschis) şi spre dreapta este ON (închis).

Semnificația comutatoarelor este descrisă în tabelul 4.1.

4.2.3. Funcții

Funcțiile admise de acest terminal sînt funcțiile "standard", cele explicate în paragraful introductiv al capitolului.

Tasta **HERE IS** produce la apasărea ei transmiterea unui mesaj de identificare de pîna la 20 de caractere (necesită opțiunea de answerback instalată).

Tabelul 4.1. Semnificația microcomutatoarelor CENTRONICS

COMUTATOR	POZIȚIF	SEMNIFICAȚIE
A1	ON	self test
A2	OFF	LINE FEED 1 interlinie
	ON	LINE FEED 2 interlinii
A3	ON	AUTO LINE FEED
A4	ON	KSR emisie şi recepție
	OFF	RO numai recepție
A5	ON	stare inițială în recepție
	OFF	stare inițială în emisie
A6	ON	110 / 300 bps (funcție de
A7	ON	150 /1200 bps opțiunea
A8	ON	300 /2400 bps instalată fizic)
B1 B2	OFF OFF	fără paritate
	OFF ON	paritate impară
	ON OFF	paritate pară
B3 B4	ON OFF	cu ecou
	OFF ON	fără ecou
	ON ON	semi-duplex, sensul este dat de semnalul 202
B5	ON	protocol X-ON/X-OFF
B6		rezervă
B7		rezervă
B8		rezervă

Configurația de viteze livrată de producător se poate determina cu self test prin citirea caracterelor ce indică vitezele de transmisie (poziția 16 din listingul de test):

```
A6 01H 110      04H 300
A7 02H 150 sau  08H 1200
A8 04H 300      10H 2400
```

4.3. DAF 2010

Dispozitivul de afişare alfanumeric DAF 2010 este un terminal interactiv de teletransmisie date, de tip display. [29], [30]

4.3.1. Caracteristici

- Tub catodic cu diagonala de 31 cm; permite afişarea a 24 de linii cu 80 de coloane.
- Interfaţa serială asincronă sau sincronă standard CCITT V.24/EIA RS 232C (asincron - protocol X-ON/X-OFF sau IBM 2848; sincron - protocol TMM-VU, BSC3)
- Interfaţa pentru periferice locale
  - imprimanta paralelă (CP-100, RCD 9334, SD-1152, DZM-180);
  - imprimanta serială (CENTRONICS 761, RCD-9334, ISM-150,etc);
  - cititor de bandă perforată (opţional);
  - perforator de bandă (opţional);
  - unitate duală de casetă magnetică (opţional).
- Permite conexiune multidrop de tip inlănţuit (daisy-chain) putîndu-se lega 16 echipamente la un singur modem.
- Generatorul de caractere standard are 128 semne (în afară de setul ASCII mai sînt şi semne speciale pentru construire tabele şi semne de control).
- Opţional poate avea şi un generator de caractere programabile pentru 128 de semne programate.
- Regimuri de lucru defilare/paginat, protejat/neprotejat
- Caracterele pot fi afişate in 8 moduri diferite în funcţie de condiţiile de afişare selectate: video normal, video invers, subliniat, intermitent (clipitor) sau combinaţii ale acestora.
- Tastatura are 101 taste: numerice, alfanumerice, funcţionale şi programabile.
- Tastele sînt de tip AUTOREPEAT
- Combinarea tastelor SHIFT LOCK şi SHIFT determină afişarea caracterelor în felul următor:

SHIFT LOCK	SHIFT	Litere	Alte caractere
NU	NU	mari	caracterele de jos
NU	DA	mici	caracterele de sus
DA	NU	mici	caracterele de sus
DA	DA	mici	caracterele de sus

- Apasărea tastei SHIFT cînd SHIFT LOCK este în jos (indicat de LED aprins) determina ieşire din mod SHIFT LOCK.
- Terminalul permite programarea a 16 funcţii cu ajutorul tastelor PF1...PF8 respectiv SHIFT cu PF1...PF8 care reprezintă PF9...PF16.
- Tastatura are 4 taste pentru selectarea regimului de afişare: RC (SET) (set caractere programate), BL (intermitent), UN (UL) (subliniat) şi VI (VIDEO) (video invers) [iar pentru cele care nu sînt inscripţionate, sînt în ordinea dată de la stînga la dreapta după tasta BREAK].

4.3.2. Comutatoare. Programare

În stînga tastaturii sus sînt 4 LED-uri cu următoarea semnificaţie:

POWER ON	terminalul este sub tensiune
KEY LOCK	tastatura operaţională
ON LINE	terminal în legatură cu calculatorul
READY	există legătură fizica cu calculatorul

Programarea regimurilor de lucru se face în stare OFF LINE prin tastarea caracterelor **ESC C** si **ESC R**. Prin această operaţie pe primul rînd al ecranului se afişeaza un text specific. Selectarea regimului de lucru se face prin tastarea unui "*" pe cîmpul dorit (tastarea caracterului spaţiu înseamna deselectare regim). Deplasarea cursorului între cîmpuri se face cu tastele -> şi <-. Terminarea selectării regimului de lucru se face prin tastare CR (punct pentru variantele mai vechi).

Prin ESC C se selectează:

ASYN/SYN	* asincron ƀ sincron

În asincron:

CHAR/BLOCK	* asincron mod caracter ƀ asincron mod bloc
FULL/HALF	* cu ecou/fără ecou
sau LINE	* mod linie asincron ƀ mod bloc asincron
EVEN ODD	ƀ ƀ nu se transmite bit paritate ƀ * paritate impară * ƀ paritate pară * * paritate indiferentă
BD.RATE	^ se crește viteza v se scade viteza (vitezele care se pot selecta sînt: 110, 150, 300, 600, 1200, 2400, 4800 si 9600 bps)

În sincron:

STATION>nnnn<DEVICE adresa stație și periferic

LINE	* căutare START MESAJ de la început linie ƀ căutare START MESAJ de la început ecran

Prin ESC R se selectează:

PR.FIELD	* terminal în regim protejat
FIELD B	* afișare intermitent
U	* afișare subliniat
RV	* afișare video invers
SCROLL/PAGE	* regim.defilare/paginat
PROG.TAB	* tabulare programabilă de pe linia de comunicație ƀ tabulare pe tabulatori impliciți
K.L.	* tastatura blocată (se deblochează la comanda de pe linia de comunicație)
AUTO LF	* la fiecare CR se face și un LF

Prin ESC B se programează tabulatorii. Pe primul rînd al ecranului apare un rînd numerotat 0...9 de 8 ori. Marcarea tabulatorilor se face prin tastare "*" pe poziția dorită, ștergere cu spațiu iar deplasarea pe rînd se face cu tastele -> sau <-. Terminare marcare tabulatori se face cu CR.

4.3.3. Funcții

STOP TAG Tabulare înapoi

CLEAR Ștergere tot ecranul în regim OFF LINE iar în regim ON LINE se șterge condiția de eroare periferice

ERASE TO END Ștergere selectiva ecran începînd de la poziția cursorului pîna la sfîrșitul ecranului (numai zonele neprotejate)

ERASE LINE Șterge toate caracterele neprotejate de pe linie de la poziția cursorului pîna la prima poziție protejată sau pîna la sfîrșitul liniei

INS CHAR Inserează caracter - de la poziția cursorului pîna la sfîrșitul liniei, toate caracterele se deplasează o poziție la dreapta, lăsind loc pentru a introduce un caracter nou

DEL CHAR Șterge caracter - deplasarea la stînga cu un caracter a tuturor caracterelor de pe linia curentă de la poziția cursorului pîna la sfîrșitul liniei

INS LINE Inserare linie - de la poziția cursorului se deplasează toate liniile cu una în jos (ultimul rînd se pierde) în regim paginat; de la poziția cursorului se deplasează toate liniile în sus (primul rînd se pierde) în regim defilare

DEL LINE Ștergere linie - în regim paginat de la poziția cursorului se deplasează toate liniile în sus cu un rînd; în regim defilare deplasarea se face în jos

ERASE Ștergere generală în ambele regimuri ON/OFF LINE

START Start mesaj - indicator început bloc de date în cazul operațiilor de transmitere către calculator sau spre imprimantă (la lucrul în mod bloc)

PRINT Tipărire - imprimare conținut ecran pe imprimanta de recopiere

START TAG Transmite - comandă începerea transferului de date către calculator

RESET Trecere terminal în regim OFF LINE

BREAK Trecerea display-ului în mod ON LINE dacă era în mod OFF LINE sau emite semnal BREAK dacă este în mod ON LINE (utilizat doar în transmisia asincronă).

Lista semnalelor prezente la conectorul de interfață pentru conexiune multipunct

(cupla tip Cannon de 25 pini; tată)

[pentru DAF 2010]

Pin	Circuit (CCITT V.24)	Denumire	Semnificație	Sens MODEM - TERMINAL
1	101	Ground	Protective ground	
2	103	TD	Transmitted data	<-----
3	104	RD	Received data	----->
4	105	RTS	Request to send	<-----
5	106	CTS	Clear to send	----->
6	107	DSR	Data set ready	----->
7	102	GND	Signal ground	
8	109	DCD	Data carrier detect	----->
9	-			
10	-			
11	-			
11	-			
12	-			
13	-			
14	-			
15	114	TC	Transmitter clock	----->
16	-			
17	115	RC	Receiver clock	----->
18	-			
19	-			
20	108/2	DTR	Data terminal ready	<-----
21	-			
22	-			
23	-			
24	TC1*	Transmitter clock 2400, 4800, 9600 Hz		----->
25	-			

* Pinul 24 se folosește doar în regim sincron

Lista semnalelor prezente la conectorul de interfață hard copy pentru transmisia serială a datelor

(conector tip Cannon de 25 de pini; tată)

[pentru DAF 2010]

Pin	Denumire	Semnificație	Sens
1	Ground	Protective ground	
2	-		
3	TD	Transmitted data	--->
4	-		
5	-		
6	-		
7	GND	Signal ground	
8	DCD	Data carrier detect	<---
9	-		
10	-		
11	-		
12	-		
13	-		
14	-		
15	-		
16	-		
17	-		
18	-		
19	-		
20 *	DTR	Data terminal ready	<---
21	-		
22	-		
23	-		
24	-		
25	-		

Realizarea legăturilor în cablul

adaptor pentru legătura multipunct

(daisy-chain)

[pentru DAF 2010]

Conector priză 25 contacte	Legătura	Conector priză 25 contacte
1	<————>	1
2	<————>	2
3	<————>	3
4	<————>	4
5	<————>	5
6	<————>	6
7	<————>	7
8	<————>	8
Pin .		. Pin
.		.
.		.
15	<————>	15
.		.
17	<————>	17
.		.
.		.
.		.
20	<————>	20

Observație

Corespondența pinilor cu semnalele CCITT V.24 se găsește în figura 2.2.

4.4. DAF 2015

DAF 2015 este un terminal de tip display; pe aceeași structură hard se realizează 2 variante prin modificarea programelor interne; aceste variante sînt: [31]

- 1. DAF 2015-D compatibilă cu DAF-2010 asincron;
- 2. DAF 2015-V compatibilă cu VT-52 (al firmei DEC).

4.4.1. Caracteristici

- Monitor TV cu diagonala tubului cinescop de 31cm (IEP) sau 44cm (TEHNOTON)
- Tastatura are 2 variante compatibile cu DAF-2010 sau cu VT-52 și cuprinde taste:
  - alfanumerice (tip QWERTY);
  - numerice;
  - speciale;
  - funcționale.
- Interfața de comunicație cu calculatorul este serială-asincronă și respectă prescripțiile CCITT V.24
- Codul utilizator este ASCII
- Opțional se poate cupla la acest terminal o imprimantă de recopiere
- Formatul de afișare a informației este de 24 de rînduri cu 80 de caractere pe fiecare rînd. Fiecare caracter este definit printr-o matrice de 7x11 puncte
- Echipamentul are în varianta standard 94 caractere afișabile (cifre, litere mari, litere mici, semne de punctuație)
- Caracterele pot fi afișate în 8 moduri diferite în funcție de condițiile de afișare selectate (video normal, video invers, subliniat, clipitor) sau de combinații ale acestora
- Combinarea tastelor CAPS și SHIFT determina afișarea caracterelor (litere mari/mici, caracterele de pe tastele cu dublă inscripționare) în modul urmator:

CAPS.	SHIFT	LITERE	ALTE CARACTERE
NU	NU	mici	caracter de jos
NU	DA	mari	caracter de sus
DA	NU	mari	caracter de jos
DA	DA	mari	caracter de sus

- In varianta DAF 2015-D tastele sînt de tip AUTOREPEAT, iar în varianta DAF-2015-V există o tastă specială REPEAT
- In afară de indicatorul luminos CAPS mai există:

ON LINE - indică stabilirea legăturii cu calculatorul;
AUTOCOPY - dialogul cu calculatorul apare şi la imprimantă.

4.4.2. Comutatori. Programare

Tastatura conţine şi comutatori (sus în partea dreaptă) pentru stabilirea parametrilor funcţionali. Semnificaţia comutatorilor pentru varianta DAF-2015-D se prezintă în tabelul 4.2. iar pentru varianta DAF-2015-V în tabelul 4.3. (numerotarea comutatorilor se face de sus în jos) (OFF înseamna comutator spre dreapta)

Tabelul 4.2. Semnificaţia comutatorilor DAF 2015-D

Nr. comutator	Poziţia			Regim de lucru
1		ON		cu ecou
		OFF		fără ecou
2		ON		defilare
		OFF		paginat
3		ON		auto LF
		OFF		fără auto LF
4 5 6	OFF	OFF	OFF	75 bps
	OFF	OFF	ON	110 bps
	OFF	ON	ON	300 bps
	OFF	ON	ON	600 bps
	ON	OFF	OFF	1200 bps
	ON	OFF	ON	2400 bps
	ON	ON	OFF	4800 bps
	ON	ON	ON	9600 bps
7 8	OFF	OFF		fără bit paritate
	OFF	ON		paritate pară
	ON	OFF		paritate impară
	ON	ON		paritate indiferentă

Tabelul 4.3. Semnificaţia comutatorilor la DAF 2015-V

Nr. comutator	Poziţia			Regim de lucru
0			ON	on line
			OFF	off line
1 2 3	OFF	OFF	OFF	75 bps
	OFF	OFF	ON	110 bps
viteza	OFF	ON	OFF	300 bps
de	OFF	ON	ON	600 bps
emisie	ON	OFF	OFF	1200 bps
	ON	OFF	ON	2400 bps
	ON	ON	OFF	4800 bps
	ON	ON	ON	9600 bps
4 5 6	OFF	OFF	OFF	75 bps
	OFF	OFF	ON	110 bps
	OFF	ON	OFF	300 bps
viteza	OFF	ON	ON	600 bps
de	ON	OFF	OFF	1200 bps
seceptie	ON	OFF	ON	2400 bps
	ON	ON	OFF	4800 bps
	ON	ON	ON	9600 bps
7 8	OFF	OFF		fără bit paritate
	OFF	ON		paritate pară
	ON	OFF		paritate impară
	ON	ON		bit paritate =1

Dupa cum se observă din tabelul 4.2. varianta DAF 2015-D poate lucra în urmatoarele moduri:
- mod defilare sau mod pagină)
- mod auto LF sau nu;
- on/off line;
- cu/fără ecou.

Varianta DAF 2015-V lucrează numai în mod:
- defilare;
- fără auto LF;
- on line/off line;
- cu ecou.

4.5. DAF 2020

DAF 2020 este un terminal grafic şi alfanumeric de tip display. [32]

În regim alfanumeric emulează un subset VDT 100 iar în regim grafic emulează TEKTRONIX 4010.

4.5.1. Caracteristici

- Unitatea centrală conţine un microprocesor Z80 şi 4Ko PROM (extensibilă la 10 Ko PROM)
- Memorie ecran 32 Ko RAM
- Tub catodic cu diagonala de 31 cm care permite afişarea a 24 linii şi 85 coloane în regim alfanumeric şi 512*288 puncte în regim grafic.
- Interfaţa cu calculatorul este bidirecţională, serială standard CCITT V.24/EIA RS 232C.
- Interfaţa serială compatibilă CCITT V.24/EIA RS 232C unidirecţională cu imprimantă de recopiere.
- Interfaţă paralelă bidirecţională pe 8 biţi pentru echipamente ca joystick, cititor/perforator bandă etc.
- Are 4 regimuri de lucru
  - alfanumeric (afişare caractere alfanumerice);
  - grafic (trasare vectori);
  - introducere grafică (de la display utilizînd cursorul cruce se transmite poziţia acestuia către calculator);
  - copie imprimantă (opţional pentru copierea imaginii de pe ecran pe o imprimantă grafică).
- Tastatura are 81 de taste şi 4 indicatoare luminoase. Indicatoarele luminoase sînt:

  LOCAL terminal necuplat la calculator
  ON LINE terminal cuplat la calculator
  KBUSY tastatura neoperaţională
  CAPS tasta CAPS este apasată
- Tastele sînt de tip AUTOREPEAT.
- Combinarea tastelor CAPS şi SHIFT determină afişarea caracterelor în modul urmator:

CAPS	SHIFT	Litere	Alte caractere
NU	NU	mici	caractere de jos
NU	DA	mari	caractere de sus
DA	NU	mari	caractere de jos
DA	DA	mari	caractere de sus

4.5.2. Comutatoare. Programare

Terminalul DAF 2020 are 14 microcomutatoare pentru programarea regimurilor de lucru. Blocul de microcomutatoare se găseşte pe tastatură în poziţia din dreapta sus. Blocul este aşezat pe orizontală, microcomutatoarele fiind numerotate, de la stînga la dreapta, de la 14 pîna la 1. Un microcomutator este pe OFF dacă este apăsat înspre jos, respectiv pe ON dacă este apăsat spre sus.

4.5.3. Funcţii

Modul de execuţie a funcţiilor depinde de regimul de lucru în care se află terminalul. Regimul de lucru ON/OFF LINE poate fi stabilit doar de operator. Restul regimurilor pot fi stabilite şi de către operator şi de către calculator.

În regim de lucru ON LINE apăsarea unei taste implică transmiterea codului corespunzător pe linia de comunicaţie. Excepţie fac funcţiile PAGE şi RESET.

Funcţiile terminalului sînt:

PAGE
- ştergе ecranul cu poziţionare cursor la început ecran;
- trecere din regim grafic în regim alfanumeric;
- eliberează bufferele de comunicaţie şi de tastatură.

RESET
- trecere în regim alfanumeric cu poziţionarea cursorului la început ecran fără ştergere ecran;
- eliberează bufferele de comunicaţie şi de tastatura.

ENTER are acelaşi efect ca şi CR.

PF1 tasta inefectivă.

PF2 folosită pentru comutare din regim defilare cu salt în regim defilare lîna (umană).

SCROLL la prima apăsare a tastei se intră în mod defilare; la următoarea apăsare se intră în mod "opreşte ecran" (se opreşte execuţia oricărei comenzi în momentul completării ultimului rînd al ecranului); din acest mod se iese prin apăsare SCROLL, PAGE sau RESET.

În regim alfanumeric terminalul poate lucra în două moduri:

- mod pagină (după pornirea terminalului, după PAGE sau RESET); la comanda LF pe ultima linie cursorul se mută pe prima linie;
- mod defilare (după apăsarea tastei SCROLL).

<u>Intrarea în regim alfanumeric se face astfel:</u>

. din regim grafic
  - din linia de comunicaţie cu US, CR, ESC FF;
  - de la tastatura cu CTRL ?, CTRL _, CR, ENTER, CTRL M, PAGE, RESET sau ESC CTRL L
  (diferenţa între diferitele moduri de trecere în regim alfanumeric constă în poziţia cursorului după trecere [cursor stînga sus, stînga jos sau capătul stîng al liniei pe care se află la ieşirea din regim alfanumeric]);

. din regim copie imprimantă
  - cu PAGE sau RESET de la tastatură;

. din regim introducere grafică
  - din linia de comunicaţie cu CR, ESC FF sau LF, VT, HT, BS, BEL dacă terminalul era în regim alfanumeric înainte de trecerea în regim introducere grafică (regim GIN);
  - de la tastatură cu PAGE sau RESET.

<u>Ieşirea din regim alfanumeric</u>
  - din linia de comunicaţie cu GS, ESC SUB, ESC ETB;
  - de la tastatură cu CTRL], ESC CTRL Z sau ESC CTRL W.

În regim grafic spaţiul de lucru este 512*480 puncte iar rezoluţia de afişare este de 512*288 puncte.

După pornire, vectorii sînt de tipul linie continuă. Tipul vectorilor poate fi schimbat cu comanda CTRL A de la tastatură sau cu SOH din linia de comunicaţie, urmată de 2 caractere ASCII ai căror ultimi 4 biţi, concatenaţi, dau 8 biţi ce indică modelul liniei tastate (linie continuă, linie punctată, linie întreruptă).

<u>Intrarea în regim grafic se face astfel:</u>

. din regim alfanumeric cu GS din linia de comunicaţie sau cu CTRL] de la tastatură;

. din regimul de introducere grafică numai prin comenzi din linia de comunicaţie: cu GS dacă terminalul fusese în regim grafic înainte de intrarea în GIN, cu orice caracter ASCII diferit de CR şi ESC FF.

<u>Ieşirea din regim grafic</u>
  - din linia de comunicaţie cu CR, US, ESC FF, ESC SUB, ESC ETB;
  - de la tastatura cu una din comenzile PAGE, RESET, ESC CTRL L, ESC CTRL X, ESC CTRL W, CR, CTRL M, CTRL _, CTRL ?.

La intrarea în regim grafic, punctul grafic este cel de la ultima ieşire din mod grafic sau, după punerea sub tensiune a terminalului, PAGE sau RESET X=0, Y=0. Primul grup de coordonate primit după intrarea în regim grafic are ca efect doar schimbarea punctului grafic nu şi trasarea de vectori.

Informaţiile de trasare a vectorilor (coordonatele acestora) se transmit către calculator prin secvenţe de cîte 4 caractere ASCII.

Procedeul de aflare a coordonatelor pentru fiecare punct, cît şi alte detalii privind acest terminal, se găsesc în "Cartea tehnica DAF 2020" editat de FEPER în 1984.

În regim grafic tastele de poziţionare cursor au următoarea semnificaţie:

^ / ↓ - deplasare linie orizontală din cursor cruce în sus/jos (cu SHIFT apăsat cu un punct, în caz contrar cu 12 puncte)
- la apăsarea simultană a două taste se obţine următoarea deplasare a liniei orizontale a cursorului cruce:
  ^ <- stînga sus   ^ -> dreapta sus
  ↓ <- stînga jos   ↓ -> dreapta jos

<- / -> - deplasare linie verticală din cursor cruce spre stînga/dreapta (cu SHIFT apăsat cu un punct, în caz contrar cu 12 puncte)
- la apăsarea simultană a două taste se obţine următoarea deplasare a liniei verticale din cursorul cruce:
  <- ^ stînga sus   <- ↓ stînga jos
  -> ^ dreapta sus   -> ↓ dreapta jos

Regimul introducere grafică (GIN).

Prin software, calculatorul poate primi următoarele informaţii pentru introducerea grafică:

1. Coordonatele poziţiei cursorului alfa
2. Coordonatele poziţiei cursorului cruce
   a) activează cursorul cruce şi primeşte poziţia actuală a acestuia;
   b) activează cursorul cruce şi aşteaptă ca operatorul să schimbe poziţia acestuia în punctul dorit; cînd operatorul tastează un caracter, coordonatele poziţiei sînt trimise la calculator.

Intrarea în regim introducere grafică se face tastînd ESC CTRL Z iar ieşirea din acest regim se face cu PAGE după care terminalul intră automat în regim alfanumeric.

Tabelul 4.4. Semnificație comutatoare DAF 2020

COMUTATOR	POZIȚIE	SEMNIFICAȚIE
S3 S2 S1	OFF OFF OFF	Viteza 150 bps
	ON OFF OFF	Viteza 300 bps
	OFF ON OFF	Viteza 600 bps
	ON ON OFF	Viteza 1200 bps
	OFF OFF ON	Viteza 2400 bps
	ON OFF ON	Viteza 4800 bps
	OFF ON ON	Viteza 9600 bps
	ON ON ON	Viteza 19200 bps
S5 S4	OFF OFF	fără paritate
	ON OFF	paritate indiferentă
	OFF ON	paritate impară
	ON ON	paritate pară
S6	OFF	un bit de stop
	ON	doi biți de stop
S7	OFF	caracter de 7 biți
	ON	caracter de 8 biți
S8	OFF	regim OFF LINE
	ON	regim ON LINE
S10 S9	OFF OFF	nu se transmite nimic
regim de	ON OFF	nu se transmite nimic
introducere	OFF ON	se transmite CR
grafica	ON ON	se transmite CR și EOT
S11	OFF	transmisie fără ECOU
	ON	transmisie CU ECOU
S12	OFF	numai LF
	ON	LF plus CR
S13	OFF	coordonate întregi
rg.grafic	ON	coordonate divizate cu 2
S14	OFF	apăsarea unei taste nu este însoțita de semnal sonor
	ON	la apăsarea unei taste se emite un semnal sonor

4.6. VDT 40C

VDT 40C este un terminal alfanumeric de tip display. [33]

4.6.1. Caracteristici

- Tub catodic cu diagonala de 31 cm pe care se pot afișa 24 de linii cu 80 de coloane
- Interfața serială cu calculatorul, standard CCITT V.24/EIA RS 232C
- Regim de lucru cu sau fără ecou
- Interfață serială pentru imprimantă, unidirecțională, standard CCITT V.24/EIA RS 232C
- Mod de afișare video invers, subliniere și intermitent
- Regim de defilare bidirecțională
- Regim de lucru alfanumeric
- Tastatura are 69 de taste alfanumerice și funcționale
- Are tastă de REPEAT
- Combinarea tastelor CAPS LOCK și SHIFT determină afișarea caracterelor în modul următor:

CAPS LOCK	SHIFT	Litere	Alte caractere
NU	NU	mici	caractere de jos
NU	DA	mari	caractere de sus
DA	NU	mari	caractere de jos
DA	DA	mari	caractere de sus

4.6.2. Comutatoare. Programare

Comutatoarele de configurare sînt grupate în două blocuri de microcomutatoare aflate în nișa de pe panoul posterior al terminalului, în partea de jos.

Numerotarea microcomutatoarelor este 1...8 in blocul din stinga si 9...16 in blocul din dreapta.

Tabelul 4.5. Semnificația comutatoarelor terminalului VDT 40C

COMUTATOR	POZIȚIE	SEMNIFICAȚIE
1	OFF ON	cu ecou fără ecou
2	OFF ON	blocare în coloana 80 auto LF (local) în coloana 80
3	OFF	se emit secvențele de ESCape care realizează funcțiile înscrise pe taste
	ON	se emite un singur cod din cele 33 coduri de control disponibile pentru taste
4	OFF ON	CR fără auto LF auto LF la CR
5	OFF ON	cursor subliniere cursor bloc invers
6	OFF ON	paritate pară paritate impară
7	OFF ON	paritate activată paritate dezactivată
8	OFF ON	funcționare normală funcționare în test
9 10 11 12 viteza de transmisie	ON OFF ON OFF	viteza 50 bps
	OFF ON ON OFF	viteza 75 bps
	ON ON ON OFF	viteza 110 bps
	OFF OFF OFF ON	viteza 150 bps
	ON OFF OFF ON	viteza 200 bps
	OFF ON OFF ON	viteza 300 bps
	ON ON OFF ON	viteza 600 bps
	OFF OFF ON ON	viteza 1200 bps
	ON OFF ON ON	viteza 1800 bps
	OFF ON ON ON	viteza 2400 bps
	ON ON ON ON	viteza 4800 bps
13 14 15 16 viteza de recepție	la fel ca viteza de transmisie	aceleași viteze ca la viteza de transmisie

4.6.3. Funcții

Pe tastatură se găsesc tastele funcționale. Efectul folosirii lor este următorul (pentru cele diferite de cele generale descrise în paragraful introductiv):

ERASE TO END șterge ecran de la poziția curentă a cursorului pînă la sfîrșit ecran

ERASE LINE șterge linie curentă

PRINT tipărire conținut ecran pe imprimanta de recopiere

REP repetare caractere la apăsarea simultană cu altă tastă

START pentru a cere un nou ecran de informații de la calculator în modul "reține ecran"

ERASE pentru a cere o noua linie de informații de la calculator în modul "reține ecran"

TAB tabulatorii sînt în coloanele 1, 9, 17, 25, 33, 41, 49, 57, 65, 73 (prima coloană are numărul 1 iar ultima 85)

Terminalul VDT 40C are comenzi specifice de poziționare cursor și selectare mod de lucru, care sînt obținute prin secvențe de ESCape:

ESC B cursor în jos - dacă cursorul se află pe ultima linie nu se execută "defilare în sus" ci: a) cursorul nu se mută dacă comutatorul 2 (C2) este deschis; b) cursorul se mută pe prima linie dacă C2 este închis

ESC I cursor în sus - dacă cursorul se află pe prima linie, nu se mută, dar toată informația de pe ecran se mută în jos cu o linie (ultima linie se pierde) - "defilare în jos"

ESC A cursor în sus - cînd cursorul se află pe prima linie nu se mută dacă C2 este deschis; sau se mută pe ultima linie dacă C2 este închis

ESC C cursor la dreapta - dacă se află în ultima coloană, cursorul rămîne pe loc dacă C2 este deschis sau se mută în prima coloană a liniei următoare dacă C2 este închis; cînd cursorul se află în ultima coloană a ultimei linii și C2 este închis are loc o "defilare în sus"

ESC D cursor la stînga - dacă se află în prima coloană cursorul rămîne pe loc dacă C2 este deschis sau se mută în ultima coloană a liniei anterioare dacă C2 este închis; dacă cursorul este pe prima coloană a primei linii și C2 este închis are loc o "defilare în jos"

ESC H reset cursor - deplasare cursor în poziția inițială (prima linie prima coloană)

ESC Y poziționare absolută cursor - următoarele două caractere dupa ESC Y se interpretează ca număr linie și ca număr coloană și determină deplasarea cursorului pe linia și coloana specificată

ESC E <u>ştergere ecran</u> - ştergere ecran şi poziţionare cursor la început ecran

ESC J <u>ştergere ecran</u> - ştergere ecran de la poziţia cursorului pîna la sfîrşitul ecranului

ESC K <u>ştergere linie</u> - ştergere de la poziţia cursorului pîna la sfîrşitul liniei

ESC L <u>inserare linie</u> - toată informaţia din linia pe care se află cursorul pînă la sfîrşitul ecranului se mută cu o linie în jos (ultima linie se pierde) şi în locul liniei curente se insereaza o linie vidă

ESC M <u>ştergere linie</u> - toată informaţia de la linia următoare celei pe care se află cursorul pîna la sfîrşitul ecranului se mută în sus cu o linie (linia pe care se află cursorul se pierde), ultima linie devine o linie vidă

ESC Z <u>identificare tip terminal</u> - terminalul transmite ca răspuns o secvenţa de ESCape de trei caractere care dă calculatorului următoarele informaţii: a) terminal pornit, conectat la calculator şi răspunde la comenzi; b) este un terminal VDT 40C compatibil VT 52 fără hard copy

ESC [<u>activare mod reţine ecran</u> - datele nu vor părăsi ecranul fără confirmarea operatorului dată prin apăsarea tastelor ERASE sau START

ESC \ <u>dezactivare mod reţine ecran</u> - se permite defilarea datelor pentru a face loc pentru alte date venind de la calculator

ESC F <u>activare mod grafic</u> - la recepţia codurilor din domeniul 5E - 7E, caracterele vor fi convertite în simboluri speciale înainte de a fi afişate pe ecran

ESC G <u>dezactivare mod grafic</u>

ESC = <u>activare mod alternat</u> - terminalul intră în mod alternat pentru grupul de taste funcţionale, în care tastele ERASE LINE, ERASE TO END şi HOME transmit secvenţe de ESCape

ESC > <u>dezactivare mod alternat</u>

ESC N <u>iniţiere cîmp video invers</u>

ESC W <u>iniţiere cîmp video intermitent</u>

ESC O <u>terminare cîmp video invers/intermitent</u>

4.7. RCD 9334

Terminalul RCD 9334 este de tip receive only, uzual denumit SCAMP. [13]

4.7.1. Caracteristici

. RCD 9334 este o imprimantă cu impact (capul de imprimare este o matrice din puncte)
. Lucrează la viteza de 150 de caractere pe secundă
. Tiparirea este bidirecţională şi este controlată de un microprocesor Z80
. O linie are maximum 136 caractere
. Imprimanta se poate livra cu una din interfetele:
  - paralelă (compatibilă CENTRONICS);
  - serială (standard CCITT V.24/EIA RS 232C) care operează într-unul din protocoalele X-ON/X-OFF, BUFFER BUSY sau CHARACTER BUSY.
. Bufferul intern al imprimantei are 256 caractere.

4.7.2. Comutatoare. Programe.

Comutatorul de ON/OFF se găseşte pe panoul din spate în stînga.

Are 4 comutatori/indicatori pe panoul din faţă, în dreapta cu următoarea funcţie:

HOLD comutator apăsat indică imprimanta OFF LINE iar ridicat imprimanta ON LINE (indicatorul luminos este aprins în ambele cazuri)

RESET la apăsarea acestui comutator se resetează indicatorii de stare a imprimantei (linia curentă fiind memorată ca poziţie început pagina); indicatorul luminos aprins indică lipsa hirtie la imprimantă

STEP la apăsarea comutatorului se produce avansul hîrtiei (avansul este continuu pe rînduri atîta timp cît STEP este apăsat)

FORM la apăsarea comutatorului se produce avansul hîrtiei pe pagini (o apăsare înseamna o pagina avans, cît timp se ţine apăsat se sar pagini)

Cu HOLD apăsat, la apăsarea simultană STEP şi FORM se începe secvenţa de auto test (imprimarea în continuu a întregului set de caractere). În momentul ridicării comutatorului HOLD auto testul este oprit.

Imprimanta dotată cu interfață serială are trei blocuri de microcomutatoare poziționate pe panoul din spate în partea stîngă. Blocurile sînt poziționate astfel: unul sus (B) și două jos deoparte și de alta a blocului de sus (A în stînga și C în dreapta).

Tabelul 4.6.

COMUTATOR	POZIȚIE	SEMNIFICAȚIE
A1	ON	conectare semnal pămînt
A2	ON	recepție date pe pin 2
A3	ON	recepție date pe pin 3
A4	OFF	X-ON/X-OFF activat
A5	ON	+5V pe pin 4 (RTS)
A6	ON	+5V pe pin 5 (DTR)
A7	ON	X-ON/X-OFF transmis pe pin 2
A8	ON	X-ON/X-OFF transmis pe pin 3
B1	ON	viteza 9600 bps
B2	ON	viteza 4800 bps
B3	ON	viteza 2400 bps
B4	ON	viteza 1200 bps
B5	ON	viteza 300 bps
B6	ON	viteza 110 bps
B7	ON	1 bit STOP
	OFF	2 biți STOP
B8	ON	caracter pe 7 biți
	OFF	caracter pe 8 biți
C1	ON	BUSY pe pin 11
C2	ON	BUSY pe pin 19
C3	ON	BUSY pe pin 4
C4	ON	BUSY pe pin 20
C5	ON	CHARACTER BUSY -
C6	ON	CHARACTER BUSY +
C7	ON	BUFFER BUSY -
C8	ON	BUFFER BUSY +
C9	ON	+5V pe pin 19 (SRTS)
C10	ON	+5V pe pin 11

Imprimantele cu interfață paralelă au un singur bloc de microcomutatoare, situat pe panoul posterior al imprimantei în partea stînga. Microcomutatoarele sînt numerotate de la stînga la dreapta cu 1...8. Poziția apăsat în sus reprezintă comutator ON.

Tabelul 4.7.
Semnificația comutatoarelor imprimantei RCD 9334 cu interfața paralela

COMUTATOR	POZIȚIE	SEMNIFICAȚIE
C1 C2	ON OFF	Detectare lumină
C3 C4	ON OFF	Defect
C5 C6	ON OFF	Imprimantă selectată
C7 C8	ON OFF	Lipsa hîrtie

Imprimanta RCD 9334 are şi un set de microcomutatoare interne, cu care se programează setul implicit de lucru al terminalului. Microcomutatoarele se găsesc pe placa logică şi sînt numerotate cu 1...10 (de la stînga la dreapta dinspre carul de imprimare). (ON comutator apăsat spre panoul posterior, OFF înspre utilizator).

Tabelul 4.8. Semnificaţie comutatoare interne la RCD 9334

COMUTATOR	POZIŢIE	SEMNIFICAŢIE
1	ON	format 80 car/linie (la 10 cpi)
		119 car/linie (la 13.6 cpi)
		132 car/linie (la 16.5 cpi)
	OFF	format 136 car/linie (10 cpi)
		189 car/linie (la 13.6 cpi)
		230 car/linie (la 16.5 cpi)
2	ON	selectează set alternat de caractere
	OFF	selectează set standard de caractere
3	ON	hîrtie de 12 inch lungime (standard în ţară)
	OFF	hîrtie de 11 inch lungime
4	ON	8 linii/inch (LF face şi CR)
	OFF	6 linii/inch (LF face şi CR)
5	ON	cu AUTO LF
	OFF	fără AUTO LF
6	OFF	complementare cod caracter faţă de 1
7	ON	filtrează CR
8	ON	autoperforaţie (autoskip)
9 10	OFF OFF	10 cpi
	OFF ON	16,5 cpi
	ON OFF	13,6 cpi
	ON ON	10 cpi (dublă densitate)

Configuraţia standard este: comutatorii 3 si 6 pe ON iar restul pe OFF.

4.7.3. Funcţii

Comenzile pe care le execută imprimanta sînt descrise în tabelul următor:

Tabelul 4.9. Funcţii RCD 9334

COD	SEMNIFICAŢIE
BEL	alarmă
LF	LINE FEED
VT	tabulare verticală
FF	pagină nouă
CR	CARRIAGE RETURN
SO	început text cu caractere elongate
SI	sfîrşit text cu caractere elongate
RS	încarcare format VFU
DLE	EVFU canal 1
DC1	EVFU canal 2
DC2	EVFU canal 3
DC3	EVFU canal 4
DC4	EVFU canal 5
NAK	EVFU canal 6
SYN	EVFU canal 7
ETB	EVFU canal 8
CAN	EVFU canal 9
EM	EVFU canal 10
SUB	EVFU canal 11
ESC 4	6 linii/inch
ESC 5	8 linii/inch
ESC 6	10 caractere/inch (standard)
ESC 7	16.5 caractere/inch
ESC 8	13.6 caractere/inch
ESC 9	10 caractere/inch (dublă densitate)
ESC @	set caractere standard
ESC A	set caractere alternate
ESC B	deplasare cap de scriere în sus cu 1/2 înălţime caracter
ESC C	deplasare cap de scriere în jos cu 1/2 înălţime caracter
ESC R	reveni · condiţii iniţiale [illegible] lara:e implicit)

EVFU (Electronic Vertical Format Unit) are acelaşi rol ca şi banda pilot.

4.8. RCD 9335

Imprimanta alfanumerică şi grafică uzual denumită SCAMP grafic. [19]

4.8.1. Caracteristici

- Interfaţa paralelă compatibilă CENTRONICS şi o interfaţă serială asincronă standard CCITT V.24/EIA RS 232C
- Protocol X-ON/X-OFF sau BUFFER BUSY
- Set de 95 caractere alfanumerice standard ASCII
- Posibilitatea utilizării unui set alternat de caractere (român, german, suedez, etc.)
- LINE FEED automat la fiecare CARRIAGE RETURN
- Cap de imprimare bidirecţional cu matrice de caractere de 9/9 puncte
- Viteza de tipărire este de 165 caractere pe secundă
- Facilităţi de scriere indici, puteri şi caractere elongate
- Buffer intern de memorare de 3400 caractere

4.8.2. Comutatoare. Programare

RCD 9335 are trei blocuri de cîte 4 microcomutatoare situate în exterior, pe panoul superior în stînga sus (de la stînga la dreapta sînt notate A, B şi C, numerotate de la 1...4). Un comutator este pe ON cînd este apăsat înspre utilizator.

Semnificaţia comutatoarelor externe este descrisă în tabelul 4.10.

Imprimanta are 5 indicatoare luminoase şi 5 comutatoare pe panoul frontal, în dreapta.

Indicatoarele luminoase au următoarea semnificaţie dacă sînt aprinse (de la stînga la dreapta): imprimanta ON LINE, fără hîrtie, imprimanta selectată, început pagina şi imprimantă sub tensiune (POWER ON).

Comutatoarele sînt:

HOLD comutator apăsat indică imprimanta OFF LINE iar ridicat imprimanta ON LINE.

RESET la apăsarea acestui comutator se resetează indicatorii de stare a imprimantei (linia curentă fiind memorată ca poziţie început pagina);

STEP la apăsarea comutatorului se produce avansul hîrtiei (avansul este continuu pe rînduri atîta timp cît STEP este apăsat)

FORM la apăsarea comutatorului se produce avansul hîrtiei pe pagini (o apăsare înseamna o pagină avans; cît timp se ţine apăsat se sar pagini)

TEST apăsarea implică tipărirea textului de test; o nouă apăsare a comutatorului opreşte modul test.

Tabelul 4.10. Semnificaţie comutatori externi

COMUTATOR	POZIŢIE	SEMNIFICAŢIE
A1 A2 A3 A4	ON ON ON ON	3.0 inch/pagină
lungime	OFF ON ON ON	3.5 inch/pagină
pagină	ON OFF ON ON	4.0 inch/pagină
(in inch)	OFF OFF ON ON	4.5 inch/pagină
	ON ON OFF ON	5.0 inch/pagină
	OFF ON OFF ON	5.5 inch/pagină
	ON OFF OFF ON	6.0 inch/pagină
	OFF OFF OFF ON	7.0 inch/pagină
	ON ON ON OFF	8.0 inch/pagină
	OFF ON ON OFF	8.5 inch/pagină
	ON OFF ON OFF	9.0 inch/pagină
	OFF OFF ON OFF	11.0 inch/pagină
	ON ON OFF OFF	11.5 inch/pagină
	OFF ON OFF OFF	12.0 inch/pagină
	ON OFF OFF OFF	14.0 inch/pagină
	OFF OFF OFF OFF	17.0 inch/pagină
B1	OFF	6 linii pe inch
	ON	8 linii pe inch
B3	OFF	132 caractere pe linie
	ON	80 caractere pe linie
B3 B4	OFF OFF	10 caractere/inch
	ON OFF	16.5 caractere/inch
	OFF ON	12 caractere/inch
	ON ON	10 cpi (dublă densitate)
C1	OFF	interfaţă paralelă
	ON	interfaţă serială
C2	OFF	set standard de caractere
	ON	set alternat de caractere
C3	ON	interfaţă serială cu protocol X-ON/X-OFF
	OFF	protocol BUFFER BUSY
C4	OFF	set caractere standard
	ON	set caractere de tipar-NLQ

Pe placa logică, în interiorul imprimantei se găsesc 4 blocuri de microcomutatoare a căror semnificație se descrie în tabelul 4.11. (comutator ON este comutator închis) (notate de la stînga la dreapta A, B, C, D).

Comutatoarele C8, C9 și C10 sînt utilizate la programarea interfeței paralele iar comutatoarele A1...A10 și B1...B8 la programarea interfeței seriale.

Tabelul 4.11.

COMUTATOR	POZIȚIE	SEMNIFICAȚIE
A1	ON	7 biți pe caracter
	OFF	8 biți pe caracter
A2	OFF	paritate dezactivată
	ON	paritate activată
A3	OFF	paritate pară
	ON	paritate impară
A4	ON	viteza de 9600 bps
A5	ON	viteza de 4800 bps
A6	ON	viteza de 2400 bps
A7	ON	viteza de 1200 bps
A8	ON	viteza de 600 bps
A9	ON	viteza de 300 bps
A10	OFF	2 biți de STOP
	ON	1 bit de STOP
B1	ON	+BUSY selectat
B2	ON	-BUSY selectat
B3	ON	+/-BUSY pe pin 11
B4	ON	+/-BUSY pe pin 20
B5	ON	DTR pe pin 20
B6	ON	RS 232C recepție
B7	ON	recepție în buclă de curent
B8	ON	transmisie în buclă de curent
B9	ON	recepție în buclă de curent
B10	ON	recepție în buclă de curent
C4 C3 C2 C1 lungime pagină (in inch)	ON ON ON ON	3.0 inch/pagină
	ON ON ON OFF	3.5 inch/pagină
	ON ON OFF ON	4.0 inch/pagină
	ON ON OFF OFF	4.5 inch/pagină
	ON OFF ON ON	5.0 inch/pagină
	ON OFF ON OFF	5.5 inch/pagină
	ON OFF OFF ON	6.0 inch/pagină
	ON OFF OFF OFF	7.0 inch/pagină
	OFF ON ON ON	8.0 inch/pagină
	OFF ON ON OFF	8.5 inch/pagină
	OFF ON OFF ON	9.0 inch/pagină
	OFF ON OFF OFF	11.0 inch/pagină
	OFF OFF ON ON	11.5 inch/pagină
	OFF OFF ON OFF	12.0 inch/pagină
	OFF OFF OFF ON	14.0 inch/pagină
	OFF OFF OFF OFF	17.0 inch/pagină
C5	OFF	cap tipărire cu 9 ace
C6	ON	set caractere programate
C7	ON	tipărește o copie
	OFF	tipărește patru copii
C8	OFF	+Parallel Data Strob
	ON	-Parallel Data Strob
C9	OFF	-Parallel Busy
	ON	+Parallel Busy
C10	OFF	+ACK Parallel Acknowledge Strob
	ON	-ACK Parallel Acknowledge Strob
D1	OFF	6 linii/inch
	ON	8 linii/inch
D2	ON	sare 1 inch la sfîrșit de pagină
D3	OFF	selectează set standard caractere
	ON	selectare set alternat de caractere
D4	ON	permite încărcare set caractere de tipar (NLQ)
	OFF	permite încărcare set standard de caractere
D5	ON	permite XON/X-OFF cînd nu este BUSY/BUSY
D6	OFF	fără AUTO LF
	ON	cu AUTO LF
D8 D7	OFF OFF	10 cpi
	OFF ON	16.5 cpi
	ON OFF	12 cpi
	ON ON	10 cpi dublă densitate
D9	OFF	132 caractere pe linie
	ON	80 caractere pe linie
D10	OFF	interfață paralelă
	ON	interfață serială

4.8.3. Funcții RCD 9335

FUNCȚIE		SEMNIFICAȚIE
ESC 4		6 linii pe inch
ESC 5		8 linii pe inch
ESC 6		10 caractere pe inch
ESC 7		16.5 caractere pe inch
ESC 8		12 caractere pe inch
ESC 9		10 caractere pe inch (dublă densitate)
ESC @		set standard de caractere
ESC A		set alternat de caractere
ESC B		deplasare cap de scriere în sus cu 1/2 înălțime caracter
ESC C		deplasare cap de scriere în jos cu 1/2 înălțime caracter
ESC D		selectare set propriu de caractere
ESC G		intrare în mod grafic
ESC L		început încarcare set caractere
ESC R		revenire la stare standard (conform comutatoarelor interne)
ESC SP		început scriere set caractere de tipar (NLQ)
ESC		ieșire din mod scriere set caractere de tipar (NLQ)
CTRL G	BEL	alarmă
CTRL M	CR	termină linie și avans (dacă există setat AUTO LF)
CTRL J	LF	termină linie și avans
CTRL N	SO	început caractere elongate
CTRL O	SI	sfîrșit caractere elongate
CTRL ^	RS	început secvență încarcare VFU
CTRL L	FF	salt la pagină nouă (canal 1)
CTRL K	VT	tabulare verticală (canal 10)
CTRL P	DLE	EVFU canal 1
CTRL Q	DC1	selectare imprimantă
CTRL R	DC2	EVFU canal 3
CTRL S	DC3	deselectare imprimantă
CTRL T	DC4	EVFU canal 5
CTRL U	NAK	EVFU canal 6
CTRL V	SYN	EVFU canal 7
CTRL W	ETB	EVFU canal 8
CTRL X	CAN	EVFU canal 9
CTRL Y	EM	EVFU canal 10
CTRL Z	SUB	EVFU canal 11

EVFU (Electronic Vertical Format Unit) are același rol ca banda pilot.

4.9. DIAGRAM 2030

DIAGRAM 2030 este un sistem biprocesor care acoperă aria dintre un microcalculator cu afișaj grafic și un minicalculator specializat în grafica. [56]

Sistemul este destinat atît lucrului independent cît și cuplării la un calculator gazdă.

Sistemul DIAGRAM poate fi folosit pe 3 nivele utilizator:

1) folosirea DIAGRAM ca terminal grafic, pe baza unei unități centrale cu Z80 (U 880), cu 64 ko memorie internă, 2-8 unități de disc flexibil și interfețe sincrone/asincrone pentru cuplarea cu calculatorul gazdă;
2) folosirea DIAGRAM ca sistem de sine stătător, pe baza unei unități centrale microprogramate pe 16 biți, cu ciclul unei instrucțiuni de 300 ns, memorie internă de 1 Mcot, organizată în pagini de 64 kcuvi sistemul posedă, de asemenea, linii sincrone/asincrone pentru cuplarea cu alte sisteme;
3) modificarea configurației sistemului, prin cuplarea unor periferice de tip paralel (imprimantă, cititor de cartele, cititor/perforator de bandă) și/sau modificarea subsistemului de microprograme și/sau a software-lui (portabilitate maximă).

4.9.1. Caracteristici

4.9.1.1. Structura hardware

Sistemul poseda 2 magistrale, una servind procesorului de I/E (I/O BUS) iar cea de-a doua procesorului grafic (GRAPHIC BUS).

Subsistemul de I/E (I/O BUS)

Are o structură de microcalculator, construit pe baza microprocesorului Z80 (U 880) cu o memorie de 64 ko RAM, și dispune de:

- placa **IOP** : procesor de I/E cu 128 de nivele de întrerupere și 16 canale de timp real;
- placa **IOM** : memorie I/E cu 48 ko RAM si 16 ko RAM sau PROM;
- placa **FDF 01** : formatter disc flexibil compatibil IBM 3740, simplă densitate, care poate avea 2-4 unități de disc flexibil;
- placa **ASI 01** interfață asincronă serială (standard RS 232C, CCITT V.24), cu două porturi duplex avînd vitezele de 110, 150, 300, 600, 1200, 2400, 4800, 9600 bps, ceas extern;
- placa **PRI 01** : interfață paralelă, cu o ieșire și o intrare, pe 8-12 biți, putînd lucra în cod ASCII sau HOLLERITH;
- placa **DMP 01** : procesor dublu microprogramabil (cu micromemoria de 1024*48 biți) și 4 nivele de întrerupere;
- placa **DSU 01** : procesor display cu o memorie de ecran de 512*512 biți;
- placa **KSO 02** : joystick și tastatură cu 58 de taste, 11 taste digitale, 16 taste programabile, interfață RS 232C, joystick pe 2 axe;
- 1 controller de disc cartuș (CDC).

Perifericele care se pot cupla sînt:

- disc flexibil (maximum 8 unități);
- teleprinter (SCAMP) (maximum 4 unități);
- cititor și perforator de bandă;
- cititor de cartele;
- disc cartuș;
- imprimantă grafică (plotter).

Subsistemul grafic (GRAPHIC BUS)

Are la bază procesorul microprogramat pe 16 biți, avînd structura unui minicalculator. Afișarea se face pe bază de generator de vectori pe un monitor TV.

Pe fiecare magistrală pot exista maximum 19 plăci

Configurația strapurilor pe plăci

1) Placa IOP 01 (procesorul de I/E)

Microcomutatorul DS1 stabilește configurația sistemului și asignarea terminalului folosit drept consola grafică.

	COMUTATOR	POZ	SEMNIFICATIE
1	PROG2	ON	program de test
		OFF	nefolosit
2	PROG1	ON	TTY asignat la clasa de ieșire
		OFF	CRT asignat la clasa de ieșire
3	PROG0	ON	TTY asignat la clasa de intrare (port0)
		OFF	KSO asignat la clasa de intrare (port 1)
4	DIS BUS	ON	magistrala UC decuplată
		OFF	magistrala UC cuplată

Panoul operator (legat la IOP 01 prin conectorul PB) asigură, în ordine de la stînga la dreapta:

- întrerupere nemascabilă (comutator);
- vizualizarea prezenței tensiunii de alimentare (bec 220v);
- vizualizarea stării HALT a procesorului central (bec);
- RESET (comutator).

2) Placa IOM 01 (memoria de I/E)

Microcomutatorul DS1 selecteaza modulele de memorie care sînt folosite în sistem (există 4 module a 16 ko RAM selectabile separat sau 4 module a 16 ko PROM selectabile separat sau 1 banc de 1 ko PROM), iar microcomutatorul DS2 selectează/deselectează verificarea parității, semnalarea erorilor de paritate și accesul procesorului microprogramat la placa de memorie.

3) Placa ASI 01 (interfața asincronă serială)

Cuprinde 2 linii seriale asincrone care pot lucra în regim semi-duplex sau duplex. Viteza de transmisie și semnalele pentru interfața cu modemul se programeaza prin soft la portul 0 (intrarea 11/31) pentru prima linie sau la portul 1 (intrarea 12/32) pentru a doua linie.

Configurația acestui port (octetul de date citit prin adresarea sa) este următoarea:

7	6	5	4	3	2	1	0
DSR0	CTS0	DCD0	RT30	RT20	RT10	RT00	EI

unde DSR, CTS, DCD semnifică semnalele conform standardului CCITT V.24, EI este semnalul de validare a întreruperii la recepție, RT30 selecția ceasului iar RT20, RT10, RT00 codifică viteza de transmisie astfel:

RT20	RT10	RT00	Viteza
0	0	0	110 bps
0	0	1	150 bps
0	1	0	300 bps
0	1	1	600 bps
1	0	0	1200 bps
1	0	1	2400 bps
1	1	0	4800 bps
1	1	1	9600 bps

Pentru a doua linie programarea este identică.

Microcomutatorii DS0 sau DS1, pentru cele două linii respectiv, pot fixa prin hard parametrii liniei, fără a mai fi nevoie de programarea prin soft.

	COMUTATOR	POZ	SEMNIFICATIE
1	SBS	ON	1 bit de STOP
		OFF	2 biți de STOP
2	PI	ON	paritate activă
		OFF	paritate inhibată
3	WLS	ON	7 biți de date
		OFF	8 biți de date
4	PS	ON	paritate pară
		OFF	paritate impară
5	RT3	ON	ceas intern
		OFF	ceas extern
6	RT2		
7	RT1		selectează vit. de transmisie
8	RT0		

RT2	RT1	RT0	Viteza
0	0	0	110 bps
0	0	1	150 bps
0	1	0	300 bps
0	1	1	600 bps
1	0	0	1200 bps
1	0	1	2400 bps
1	1	0	4800 bps
1	1	1	9600 bps

4.9.1.2. Structura software

Sistemul DIAGRAM oferă utilizatorului o exploatare comodă în regim conversațional, a tuturor resurselor mașinii, atît în regim de lucru independent cît și ca terminal cuplat la un calculator gazdă.

Componentele sale sînt:

- Nivel 1
  - sistemul DIOS 01 (monitor);
  - asamblor Z80 (DIAS 01);
  - editor de texte (EDI 01);
  - utilitare de sistem;
  - pachet de programe de legatură cu calculatorul gazdă;
- Nivel 2
  - posedă în plus față de nivelul 1:
  - interpretor BASIC;
  - crossasamblor de microprograme (TRANSMICRO);
  - emulator de terminale grafice;
- Nivel 3
  - poate conține în plus și alte programe de dezvoltare.

4.9.2 Operare

Pentru punerea în funcțiune a sistemului se apasă comutatorul ON/OFF aflat în stînga calculatorului, așteptînd aprinderea becului care semnalizează alimentare prezentă. După aceea se apasă comutatorul ON/OFF din spatele panoului TV și apoi comutatorul ON/OFF al discului. Se introduce discul sistem în unitatea 0, se apasă RESET de la panou. În acest moment pe ecran apare mesajul "DIOS 01 MONITOR", după care operarea și luarea de contact cu calculatorul gazdă se face conform indicațiilor manualului de utilizare. Pe panoul TV există două butoane pentru reglarea luminozității și contrastului afișajului.

4.10. ISM 150

ISM 150 este o imprimantă serială matricială destinată listării datelor sau programelor. [57]

4.10.1. Caracteristici

- Formatul caracterelor este matrice de puncte 9x9·
- Are un set de 95 caractere ASCII;
- Viteza de tipărire este de 150 caractere pe secundă
- Lăţimea maximă a liniei de caractere este 13.6 inch
- Densitatea caracterelor pe orizontală este de 10, 13.6, 16.5 cpi, dublă densitate a caracterelor şi lăţime dublă a caracterelor
- Execută pînă la 4 copii plus originalul
- Lăţimea hîrtiei este de la 1.5 la 16 inch
- Lungimea hîrtiei este de 11 sau 12 inch (selectabilă prin comutatori)
- Avansul hîrtiei se face prin metoda cu tambur cu stifturi
- Viteza de avans a hîrtiei este de 4 inch pe secundă
- Densitatea de linii este 6 sau 8 linii pe inch
- Controlul avansului pe verticală se face logic pe 11 canale
- Interfaţa serială RECEIVE/TRANSMIT (conform CCITT V.24)
- Buffer pînă la 1024 caractere

4.10.2. Comutatori. Programare

Comutatorul de punere sub tensiune este basculant şi se găseşte pe panoul posterior al imprimantei, în stînga.

Pe panoul frontal în dreapta se găsesc 4 taste de operare a imprimantei:

SL

Apăsarea acestei taste, produce selectarea/deselectarea imprimantei; dacă tasta este apăsată în timpul unei operaţii de tipărire, se inhibă tipărirea şi la umplerea buffer-ului de comunicaţie se emite codul X-OFF pe interfaţă serială. La o nouă apăsare a tastei, se tipăreşte conţinutul buffer-ului de comunicaţie şi se emite codul X-ON pe interfaţa serială.

ON LINE

Apăsarea acestei taste, produce punerea/scoaterea imprimantei în/din comunicaţie cu calculatorul.

TOF

Apăsarea acestei taste, produce avansul hîrtiei pînă la începutul paginii următoare. Este funcţională cît timp imprimanta este deselectată.

LF

Apăsarea acestei taste, produce avansul hîrtiei pe verticală. Dacă se apasă această tastă mai puţin de 1/2 secunde, hîrtia va avansa 1/48 inch. Cît timp se apasă pe această tastă, hîrtia va avansa continuu. Tasta LF este funcţională cît timp imprimanta este deselectată.

Apasarea simultană a tastelor LF şi TOF, indiferent de starea imprimantei, produce resetarea imprimantei: se scoate din comunicaţia cu calculatorul şi se poziţionează capul de tipărire la marginea stîngă. Imprimanta rămine deselectată.

Pe panoul frontal al imprimantei se mai găsesc şi 4 indicatori luminoşi (deasupra tastelor de operare):

SELECT şi ON LINE

Aceşti doi indicatori arată starea imprimantei după apasarea tastelor SL, respectiv ON LINE.

ALT SET

Acest indicator arată din ce set de imagini ale caracterelor se selectează imaginile codurilor recepţionate. Cînd indicatorul este stins caracterele tiparite sint cele standard, iar cînd indicatorul este aprins caracterele tipărite sînt cele din setul alternat.

READY

Indică dacă imprimanta este gata de lucru cu calculatorul.

11010 - comanda SARME (setare mod de lucru asincron extins);

11110 - comanda SABME (setare mod de lucru echilibrat asincron extins);

10000 - comanda SIM (stabilire mod de inițializare): folosită pentru a reinițializa legătura (pentru setarea unui mod de lucru specific unei implementari particulare, care nu poate fi definit complet prin modurile de mai sus);

- răspuns RIM - cerere de inițializare (de comandă SIM) emis de stația secundară

00000 - comandă sau raspuns UI : transfer de mesaje de informație nenumerotate;

00100 - comanda UP (invitație la emisie nenumerotată): o alta modalitate de a solicita date de la stația secundară decît cea de setare a bitului P dintr-o comanda. Ofera posibilitatea de "apel in grup" pentru mai multe stații secundare reunite sub aceeași "adresa de grup";

11101 - comandă și răspuns XID (achitare a identificarii): oferă posibilitatea stațiilor cooperante de a schimba informații despre caracteristicile lor;

00010 - comanda DISC (deconectare): folosită pentru eliberarea legăturii in rețelele comutate sau pentru a avertiza stațiile secundare că este suspendată exploatarea lor; în celelalte cazuri stația secundară va trece în mod "deconectat" după receptia comenzii;

- răspuns RD (cerere de deconectare), pentru a forța stația primară să trimita un mesaj DISC;

11000 - răspuns DM (mod deconectat): răspuns al stației secundare la comenzi de stabilire a modului de lucru, dacă la recepția lor era în mod deconectat; cerere adresată primarului de a conecta legătura prin trimiterea unei comenzi de stabilire a modului de lucru;

00110 - răspuns UA (achitare numerotată de stația secundară a comenzilor de tip "setare mod de lucru", DISC recepționate): este obligatorie achitarea acestor mesaje de către stația secundară;

4.10.3. Funcții

In tabelul următor se prezintă codurile de control ale imprimantei:

COD	FUNCȚIE
BEL	alarmă sonoră
LF	LINE FEED
VT	tabulare verticală
FF	salt la pagină nouă
CR	retur de car
SO	start caractere elongate
SI	stop caractere elongate
DLE	EVFU canal 1
DC1	EVFU canal 2
DC2	EVFU canal 3
DC3	EVFU canal 4
DC4	EVFU canal 5
NAK	EVFU canal 6
SYN	EVFU canal 7
ETB	EVFU canal 8
CAN	EVFU canal 9
EM	EVFU canal 10
SUB	EVFU canal 11
RS	start încărcare condiții de salt
ESC 4	6 linii pe inch
ESC 5	8 linii pe inch
ESC 6	10 cpi
ESC 7	16.5 cpi
ESC 8	13.6 cpi
ESC 9	10 cpi dublă densitate
ESC @	set caractere standard
ESC A	set caractere opțional
ESC B	indici superiori
ESC C	indici inferiori
ESC R	resetare condiții

EVFU (Electronic Vertical Format Unit) are acelasi rol c's banda pilot.

5. PROTOCOALE

5.1. DEFINIȚIE, OBIECTIVE

Protocolul de transmisie este un set de reguli și convenții guvernînd schimbul de date pe o legătură între componenetele unui sistem de teleprelucrare (unde componentele pot fi: calculatoare, terminale, concentratoare, etc, iar legătura poate fi pe cablu, microunde, canal satelit, etc.)

Protocoalele formează o interfață logica standard între componente. [2], [8]

Un protocol trebuie sa asigure:

- cadraj:
  - delimitare date utile;
  - furnizare sincronizare;
- control eroare:
  - detectare erori, confirmare primire corectă, cereri de retransmisie mesaje incorecte;
- control inițializare:
  - stabilire legatură activă, schimb de secvențe pentru inițializare și activare (exemplu: polling);
- control flux de informație:
  - reglarea fluxului informației acceptate în funcție de posibilitățile de prelucrare;
- control legătură:
  - stabilire și terminare conexiuni;
  - control direcție transfer;
- transparență:
  - asigurarea independenței legăturii de structura datelor transmise (imposibilitatea confundării unor date utile cu caractere de control);
- tratare excepții:
  - detectare întrerupere legătură (prin mecanism de time-out), cu tratare corespunzatoare;
  - detectare secvențe ilegale.

5.2. TIPURI DE PROTOCOALE

Protocoalele de teletransmisie diferă, în primul rînd, în funcție de modul de transmisie a informației pe linii. Deosebim astfel:

- protocoale asincrone;
- protocoale sincrone.

5.2.1. Protocoale asincrone

În asincron, informația se vehiculează caracter cu caracter și fiecare caracter e încadrat cu biții de START/STOP necesari stabilirii sincronismului stațiilor. Informațiile de control apar sub formă de caractere (sau succesiuni de caractere) speciale.

Protocoalele asincrone variază de la reguli foarte simple (TTY) pîna la protocoale complicate (specifice cuploarelor IBM - de exemplu IBM 2848) cu o logică similară celei pentru transmisie sincronă.

5.2.2. Protocoale sincrone

După modul de sincronizare, protocoalele sincrone sînt:

a) orientate caracter (exemple: TMM, BSC, DDCMP, ECMA, etc);
b) orientate bit (exemple: HDLC, SDLC, BDLC, UDLC, ADCCP, etc).

După modul în care se controlează sfîrșitul de mesaj, există următoarele tipuri:

a) la protocoalele orientate caracter:

- protocoale cu control prin caractere speciale (exemple: TMM, BSC);
- protocoale cu control prin contor (byte-count) (exemplu: DDCMP);

b) la protocoale orientate bit:

- control prin flaguri (secvențe speciale de biți);

Trecerea la protocoalele orientate bit s-a făcut din cauza urmatoarelor <u>limitări ale protocoalelor orientate pe caracter:</u>

- necesitatea de a distinge caracterele de date de cele de control încarcă software-ul si hardware-ul;
- asignarea de caractere pentru control le face nedisponibile pentru codificarea datelor în transmisie normală; utilizarea tuturor caracterelor este posibilă numai în transmisia transparentă;
- nu sînt natural transparente la structura sau codarea textului;
- transparența e complicat realizată utilizînd caracterul special DLE;
- inflexibilitate pentru echipamente, toate echipamentele din rețea trebuind sa aibă același cod;
- implică multă procesare la nivel funcțional al legăturii de date;
- au natura inerent semi-duplex, deci nu se folosește eficient o linie duplex;
- printr-o unitate de transmisie trimisă se efectuează o singură funcție, deci cere număr mare de comutări ale legăturii logice;
- mecanismele de time-out implică resincronizări multiple.

5.3. DESCRIERE PROTOCOALE SINCRONE

5.3.1. BSC

Protocolul BSC (Binary Synchronous Communication) are următoarele caracteristici generale: [12], [13], [14], [15], [30], [37]

. tipul transmisiei: sincron, punct la punct sau multipunct;
. transmisie semi-duplex;
. viteze de transmisie în general între 600 si 19200 bps;
. codul utilizat poate fi ASCII sau EBCDIC;
. mod normal și transparent;
. controlul parității: CRC 16 pentru cod EBCDIC sau LRC par cu VRC impar pentru cod ASCII;
. caractere funcționale: SOH, STX, ETB, ITB, ETX (caractere funcționale de cadraj text); ACKO, ACK1, NAK (achitări mesaje); WACK, TTD (caractere de temporizare); RVI (comutare sens logic al transmisiei); EOT (sfirsit transmisie); ENQ (interogare);
. tipuri de mesaje: text, de inițializare, de achitare recepție, sfîrșit transmisie, de temporizare;
. secvență de sincronizare (minimum 2 caractere SYN) - (nSYN)
. metode de dialog: polling - selecting și la egalitate ("contention");
. utilizare:
  - BSC 1 - pentru legătură punct la punct necomutat;
  - BSC 2 - pentru legătură punct la punct comutat;
  - BSC 3 - pentru legătură multipunct necomutat;

. caractere de control:

SYN	caracter de sincronizare;
SOH	început antet (startare calcul BCC; nu este inclus in BCC);
STX	început text;
ETB	sfîrșit bloc;
ITB	sfîrșit bloc intermediar;
ETX	sfîrșit text;
ACKO	achitare pozitivă bloc par;
ACK1	achitare pozitivă bloc impar;
ENQ	interogare (cerere "mesaj");
RVI	utilizat de stația "slave" cînd dorește să intrerupă o transmisie curentă pentru a transmite stației "master" un mesaj de înaltă prioritate;
DLE	utilizat cu alte caractere pentru secvențe de control în mod transparent;
TTD	utilizat de calculator pentru a menține conectarea cu terminalul selectat, cînd acesta nu are încă mesajul dorit pregătit pentru transmisie;
WACK	utilizat de terminal pentru a indica faptul că, temporar, nu este pregătit pentru recepție;
BCC	caracter de control paritate bloc;
PAD	caracter de temporizare (asigură întîrzierea necesară modemului pentru transmisia ultimelor caractere semnificative din mesaj înainte de a bascula între stările modulație/demodulație);

. caractere de control speciale:

CU	adresă stație master (control unit);
DV	adresă stație slave (terminal);

. diagramele de mesaje pentru protocolul BSC 3 sînt descrise în figurile 5.1, 5.2, 5.3 si 5.4; in tabele 5.1, 5.2 si 5.3 se găsesc valorile pentru adresele CU si DV.

Diagramele protocolului BSC 3 de mai jos sînt în conjunctură cu funcțiile terminalului IBM 3275 (emulat pe DAF 2010).

Comenzile sint utilizate de calculator pentru a iniția operații de scriere, citire sau ștergere date din buffer-ul terminalului selectat. Comenzile pot fi înlanțuite în așa fel încît mai mult de o comanda poate fi executată fără a readresa terminalul dar, dupa fiecare comandă, direcția transmisiei pe linie este inversată pentru a transmite o confirmare sau o cerere de reluare a mesajului.

Comenzile executate de terminal sînt de trei tipuri principale:

- comenzi de tip <u>read</u> care transferă datele de la terminal (buffer sau keyboard) sau informațiile de stare spre calculator;
- comenzi de tip <u>write</u> utilizate pentru a transmite ordine și date de la calculator spre terminal;
- comenzi de tip <u>control</u> utilizate pentru a executa anumite operații la imprimantă sau terminal.

Tabela 5.1. Adresa CU (Control Unit) în procesul de poll

Nr.CU	Adresa	Nr.CU	Adresa	Nr.CU	Adresa
0	40H(20)	11	4BH(2E)	22	D6H(4F)
1	C1H(41)	12	4CH(3C)	23	D7H(50)
2	C2H(42)	13	4DH(28)	24	D8H(51)
3	C3H(43)	14	4EH(2B)	25	D9H(52)
4	C4H(48)	15	4FH(21)	26	5AH(5D)
5	C5H(45)	16	50H(26)	27	5BH(24)
6	C6H(46)	17	D1H(4A)	28	5CH(2A)
7	C7H(47)	18	D2H(4B)	29	5DH(29)
8	C8H(48)	19	D3H(4C)	30	5EH(3B)
9	C9H(49)	20	D4H(4D)	31	5FH(5E)
10	4AH(5B)	21	D5H(4F)		

Tabela 5.2. Adrese de terminal în poll (DV)

Nr.DV	Adresa	Nr.DV	Adresa	Nr.DV	Adresa
0	40H(20)	22	D6H(4F)	45	6DH(5F)
1	C1H(41)	23	D7H(50)	46	6EH(3E)
2	C2H(42)	24	D8H(51)	47	6FH(3F)
3	C3H(43)	25	D9H(52)	48	F0H(30)
4	C4H(44)	26	5AH(5D)	49	F1H(31)
5	C5H(45)	27	5BH(24)	50	F2H(32)
6	C6H(46)	28	5CH(2A)	51	F3H(33)
7	C7H(47)	29	5DH(29)	52	F4H(34)
8	C8H(48)	30	5EH(3B)	53	F5H(35)
9	C9H(49)	31	5FH(5E)	54	F6H(36)
10	4AH(5B)	32	60H(2D)	55	F7H(37)
11	4BH(2E)	33	61H(2F)	56	F8H(38)
12	4CH(3C)	34	E2H(53)	57	F9H(39)
13	4DH(28)	35	E3H(54)	58	7AH(3A)
14	4EH(2B)	36	E4H(55)	59	7BH(23)
15	4FH(21)	37	E5H(56)	60	7CH(40)
16	50H(26)	38	E6H(57)	61	7DH(27)
17	D1H(4A)	39	E7H(58)	62	7EH(3D)
18	D2H(4B)	40	E8H(59)	63	7FH(22)
19	D3H(4C)	41	E9H(5A)		
20	D4H(4D)	42	6AH(7C)		
21	D5H(4F)	43	6BH(2C)		
		44	6CH(25)		

Nota. Codul 7FH este utilizat ca adresă DV pentru poll general.

Tabela 5.3. Adresele CU în procesul de selecție

Nr.CU	Adresa	Nr.CU	Adresa	Nr.CU	Adresa
0	60H(2D)	11	6BH(2C)	22	F6H(36)
1	61H(2F)	12	6CH(25)	23	F7H(37)
2	E2H(53)	13	6DH(5F)	24	F8H(38)
3	E3H(54)	14	6EH(3E)	25	F9H(39)
4	E4H(55)	15	6FH(3F)	26	7AH(3A)
5	E5H(56)	16	F0H(30)	27	7BH(23)
6	E6H(57)	17	F1H(31)	28	7CH(40)
7	E7H(58)	18	F2H(32)	29	7DH(27)
8	E8H(59)	19	F3H(33)	30	7EH(3D)
9	E9H(5A)	20	F4H(34)	31	7FH(22)
10	6AH(7C)	21	F5H(35)		

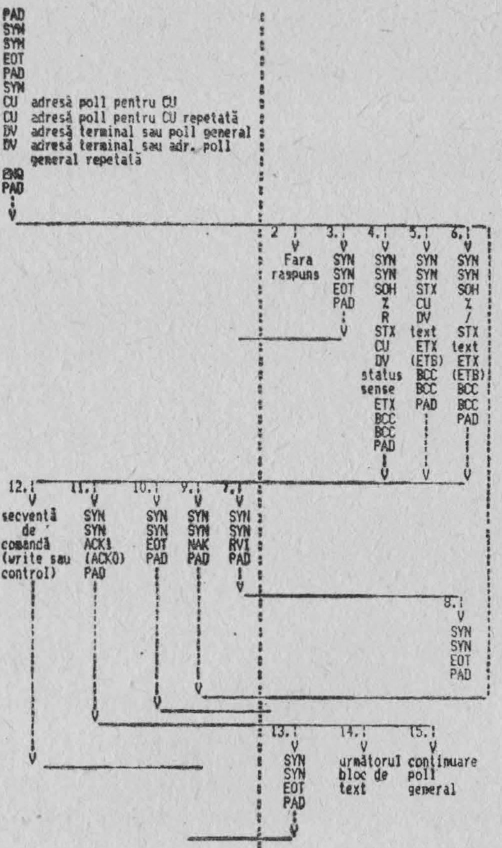


Figura 5.1. Polling general și specific, diagramă

```
CALCULATOR          TERMINAL

PAD  1.
SYN
SYN
EOT
PAD
SYN
SYN
CU   adresă de selecție CU
CU   adresă de selecție CU repetată
DV   adresă de selecție terminal
DV   adresă de selecție terminal
     repetată
ENQ
PAD
 V
        2.        3.     4.     5.
        V         V      V      V
        Fără      SYN    SYN    SYN
        răspuns   SYN    SYN    SYN
                  RVI    WACK   ACK0
                  PAD    PAD    PAD
V
SYN
SYN
EOT       V
PAD       secvență de
          de comenzi
          (vezi fig. 5.3. si 5.4.)
```

Figura 5.2. Secvență de selecție, diagramă

```
CALCULATOR          TERMINAL

V
SYN 1.
SYN
STX
ESC
CMD   cod comandă
TEXT
ETX
BCC
BCC
PAD
V
               2.        3.    4.    5.     6.
               V         V     V     V      V
               fără      SYN   SYN   SYN    SYN
               răspuns   SYN   SYN   SYN    SYN
                         NAK   EOT   WACK   ACK0
                         PAD   PAD   PAD    (ACK1)
                                            PAD
V
SYN 7.
SYN
EOT
PAD
V
9.              8.
V               V
secvență de     SYN
comenzi         SYN
adiționale      EOT
                PAD
```

Figura 5.3. Comenzi de tip WRITE și CONTROL, diagramă

```
CALCULATOR            TERMINAL
                         :
  V                      :
  SYN 1.                 :
  SYN                    :
  STX                    :
  ESC                    :
  CMD codul comenzi.     :
  ETX                    :
  BCC                    :
  BCC                    :
  PAD                    :                          read    test
  :                      :                          command request
  V                      :                                  message
                         : 2.:    3.:   5.:   6.:   7.:
                         :  V      V     V     V     V
                         : fără   SYN   SYN   SYN   SYN
                         : răspuns SYN  SYN   SYN   SYN
                         :        NAK   EOT   STX   SOH
                         :        PAD   PAD   CU    %
                         :         :     :    DV    /
                         :         V     :    text  STX
  SYN 4.                 :               V    ETX   text
  SYN                    :                    (ETB) ETX
  EOT                    :                    BCC   (ETB)
  PAD                    :                    BCC   BCC
  :                      :                    PAD   BCC
  V                      :                     :    PAD
                         :                     V     :
10.:  9.:     8.:        :                           V
  V    V       V         :
 SYN  SYN <-- SYN        :
 SYN  SYN     SYN        :
 ACK1 EOT     NAK        :
 sau  PAD     PAD        :
 ACK0          :         :
 PAD           V         :
  :                      :
  V                      : 11.:    12.:
                         :  V       V
                         : SYN     următorul
                         : SYN     bloc de
                         : EOT     text
                         : PAD
                         :  :
                         :  V
```

Figura 5.4. Comandă de tip read, diagramă

5.3.2. TMM

Protocolul TMM (Transmission Mode Message) are următoarele caracteristici generale: [17], [30], [38], [46], [47], [63]

- tipul transmisiei este sincron punct la punct sau multipunct;
- transmisie semi-duplex și duplex;
- viteze de transmisie în general între 600 si 19200 bps;
- codul utilizat este ASCII; este prevazut și codul EBCDIC, utilizat numai în mod transparent;
- mod normal și transparent;
- controlul parității: paritate încrucișată, LRC par, VRC impar; este prevazut și CRC, utilizat în mod transparent cod EBCDIC;
- caractere funcționale: SOH, STX, ETB, ETX (caractere funcționale de cadraj text), ACK, NAK, DLE ACK, DLE NAK (achitări de recepție), SEL, POL, ENQ (inițializare) și EOT (sfîrșit de transmisie);
- tipuri de mesaje: text, de inițializare, de achitare recepție, sfîrșit de transmisie;
- secvență de sincronizare (minimum 3 caractere SYN) - (nSYN)
- metode de dialog: polling - selecting și la egalitate;
- utilizare:
  - TMM-UC (unitate centrală);
  - TMM-RB (remote batch);
  - TMM-VU (consolă de vizualizare - terminal);
  - * TMM-UC și TMM-RB au numerotarea blocului în curs de transmisie inclusă în antet (caracterul NOB, vezi mai jos);
  - * în TMM-VU orice secvență de inițializare a "terminalului" este precedată de secvența nSYN EOT nSYN (această procedură a impus selectia rapidă);
- caracterele de control:
  - SYN caracter de sincronizare;
  - SOH început antet (startare calcul BCC; nu este inclus în BCC);
  - STX început text;
  - ETB sfîrșit bloc de mesaj;
  - ETX sfîrsit transmisie;
  - EOT - emis de calculator: anulează toate cererile anterioare și pune terminalul în stare așteptare a unei noi comenzi;
  - EOT - emis de terminal: indica absența postului în selecție și absența mesajelor de transmis în polling;
  - ACK achitare pozitivă;
  - NAK achitare negativă;
  - ENQ interogare (cerere de mesaj);

DLE utilizat în mod transparent numai de stația master (DLE ACK achitare pozitivă cu cerere de suspendare; DLE NAK achitare negativă cu cerere de oprire);

BCC caracter de control paritate bloc;

PAD caracter de temporizare (asigură întîrzierea necesară modemului pentru transmisia ultimelor caractere semnificative din procedură înainte de a bascula între stările modulație/demodulație);

caractere de control speciale:

SEL codul pentru selecție (valori între 60-7FH) care conține și adresa unitații de control (adrt);

AD1 identic cu SEL;

C2 conține codul funcției și adresa perifericului cu care se dorește a se face schimb de date (adrp);

AD2 identic cu C2;

POL codul pentru polling (valori între 41-5FH) în care ultimii 5 biți reprezintă adresa unitații de control (adrt);

ADR conține codul funcției și adresa post (adrp);

NOB număr de secvență bloc;

- diagramele de mesaje pentru aceste protocoale sînt:
  1. pentru TMM-UC - figura 5.5;
  2. pentru TMM-RB - figurile 5.6 si 5.7;
  3. pentru TMM-VU - figurile 5.8, 5.9, 5.10, 5.11 și 5.12;
- în implementarea protocolului TMM-RB în produsul TELESYMBIONT mesajul utilizator este prefixat de 4 caractere care reprezintă:

C2 adresa periferic solicitat la stația terminală;

C3 caracter de control special care indică dacă C4 și C5 sînt sau nu programate și dacă articolul este destinat consolei sau altui periferic;

C4,C5 caractere de control care definesc numărul de octeți al primului bloc din articol;

- la protocolul TMM-RB, în cazul recepției eronate, caracterul NOB emis cu NAK indică numărul ultimului bloc corect recepționat;
- pentru a se putea face și transfer de fișiere între calculator și terminal (terminalul fiind un mini sau un micro calculator) protocolul TMM-VU are o extensie prin care se permite comutarea, pe timpul introducerii sau extragerii unui fișier, pe un periferic la distanță; pe această durată postul este scos din lista posturilor stației master, el fiind folosit pentru dialogul de identificare fișier și dialogul de tratare a erorilor la stația slave; dialogul protocolului TMM-VU extins cuprinde în afara secvențelor clasice ale protocolului TMM-VU (fig.5.8-5.12), secvențe specifice pentru transfer de fișiere, care sînt descrise în figurile 5.18, 5.19, 5.20, 5.21 si 5.22.

În acest protocol se mai folosesc următoarele caractere de control speciale:

NOB numărul blocului - fiecare text al unui mesaj destinat unui periferic aflat la distanță sau de la un periferic aflat la distanță este precedat de un octet care indică numărul blocului; secvența de NOB-uri trebuie sa fie crescătoare și ciclică, între X'20' si X'7F'; blocurile cu același NOB se elimină, ele provenind din retransmisii;

DC1 început de articol fișier;

DC2 caractere de compactare blancuri - indică faptul că s-a compactat un număr mai mare de două blancuri și este întotdeauna urmat de un octet care reprezinta numărul de blancuri compactate;

NRB numărul de blancuri compactate;

DC3 sfîrșit fișier;

AD3 conține tipul operației cu perifericul și adresa perifericului (adrf).

În diagrame vor apare următoarele abrevieri:

NR (No Reponse)

Terminalul nu trimite mesaj de răspuns:

- cînd nu recunoaște codul de control al comunicației cerut pentru nici un mesaj;
- cind în timpul recepției mesajului s-a pierdut purtătoarea;
- cînd trebuie trimis **NAK** dar nu se recunoaște **ETX** într-un număr de caractere.

În cazul în care calculatorul trimite un mesaj de selecție sau polling și terminalul nu răspunde, CPU retransmite același mesaj.

IVB (Invalid Block)

Un bloc este invalid dacă :

- nu este încadrat corect de SOH și ETX;
- primul caracter SEL este eronat;
- cele două caractere **SEL** nu sînt identice;
- orice cod în oricare din tipurile de mesaje existente este eronat.

Deoarece terminalul nu recunoaște blocurile invalide, el nu răspunde și asteaptă retransmiterea blocului. Dacă la calculator se primește un bloc invalid, acesta trebuie să răspundă cu mesaj de achitare negativa (NAK) pe care terminalul îl interpreteaza ca o cerere de retransmisie.

ICB (Incorrect Block)

Un bloc este incorect dacă :

- al doilea caracter SEL este cu eroare de paritate;
- caracterul C2 este eronat;
- caracterele de date sînt cu erori de paritate;
- mesajul este cu eroare de paritate.

Un bloc incorect determină un raspuns NAK atît de la calculator cît și de la terminal.

ICS (Incorrect Sequence)

Lipsa unui EOT imediat înaintea caracterului SEL sau POL reprezintă o secvență incorectă. Acestea sînt singurele condiții de secvență incorectă recunoscută de terminal. Terminalul nu răspunde în acest caz.

O secvență incorectă la calculator este un mesaj ACK incorect. În acest caz, calculatorul trebuie să retransmită mesajul.

adrp

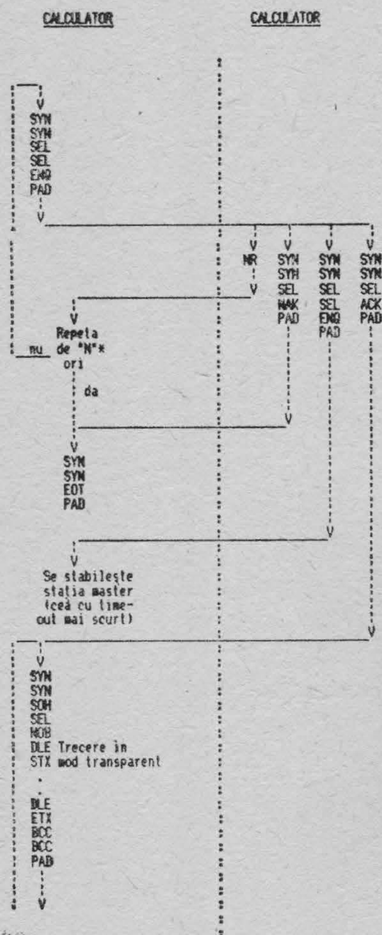
Adresa post (vezi AD2, ADR);

adrt

Adresa terminal (vezi SEL, POL);

adrf

Adresa periferic (vezi AD3).



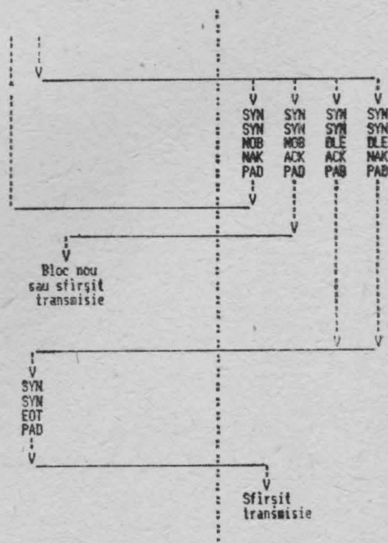


Figura 5.5. Diagrama protocolului TX4 UC

CALCULATOR TERMINAL

SYN SYN POL ADR ENQ PAD

SYN SYN EOT PAD

SYN SYN SOH SEL NOB STX . ETB BCC PAD

Stație nepregatită

Transfer bloc

SYN SYN NOB ACK PAD

SYN SYN DLE ACK PAD

SYN SYN DLE NAK PAD

SYN SYN NOB NAK PAD

SYN SYN EOT PAD

Sfîrșit transmisie

SYN SYN SOH SEL NOB STX . ETX BCC PAD

Ultim bloc

SYN SYN NOB ACK PAD

SYN SYN DLE ACK PAD

SYN SYN DLE NAK PAD

SYN SYN NOB NAK PAD

SYN SYN EOT PAD

Figura 5.6. TMM-RB. Polling

CALCULATOR TERMINAL

SYN SYN SEL ADR ENQ PAD

SYN SYN SEL NAK PAD

SYN SYN SEL ACK PAD

SYN SYN SOH SEL NOB STX . ETB BCC PAD

SYN SYN NOB NAK PAD

SYN SYN DLE ACK PAD

SYN SYN NOB ACK PAD

SYN SYN EOT PAD

Sfîrșit transmisie

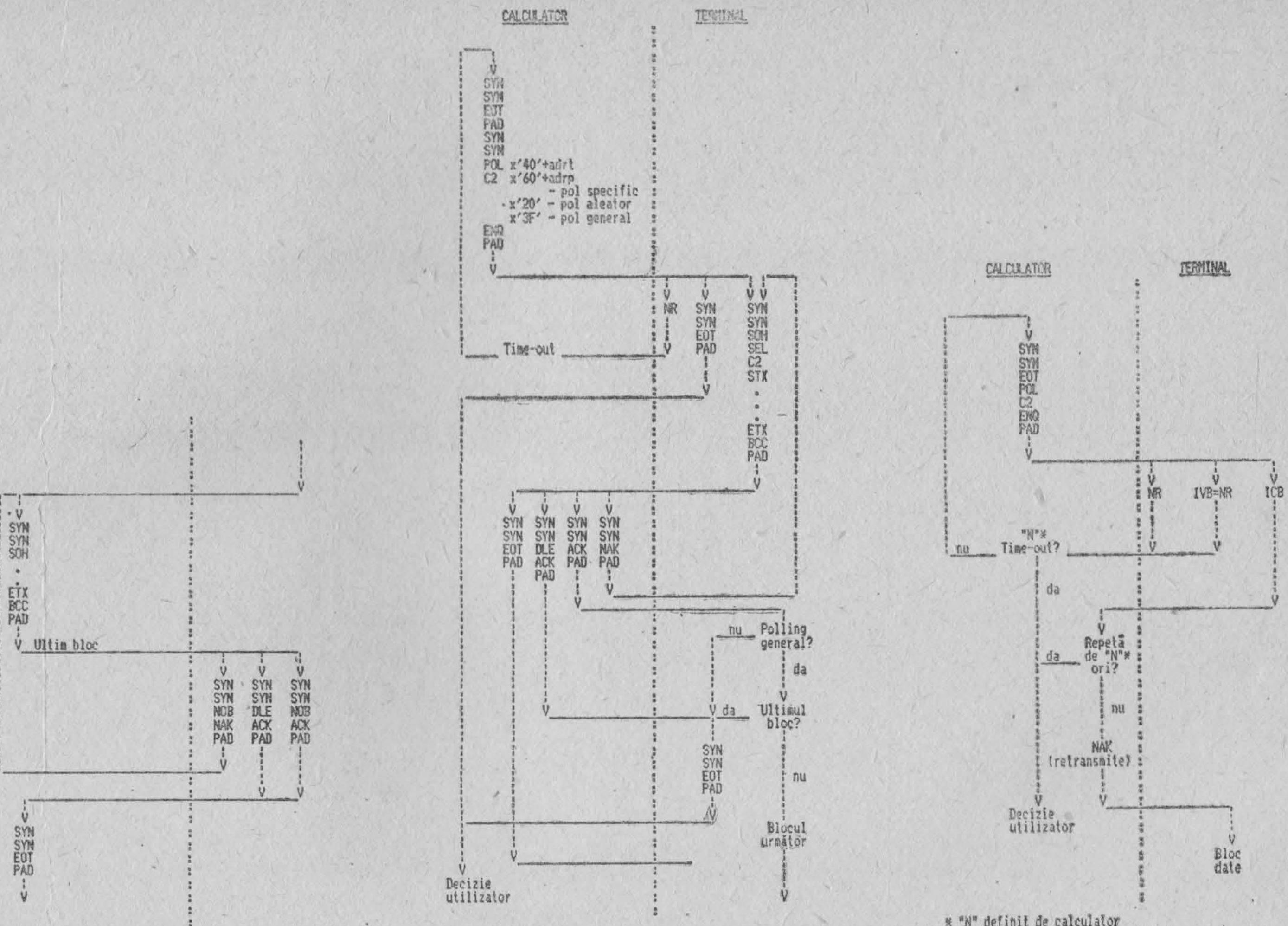


Figura 5.7. TMM-RB Selecție

Figura 5.8. TMM-VU Polling

Figura 5.9. TMM-VU Recuperare erori polling

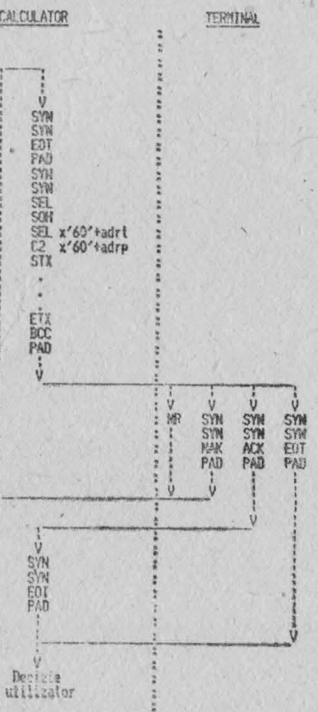


Figura 5.10. TMM-VU Selecţie rapidă

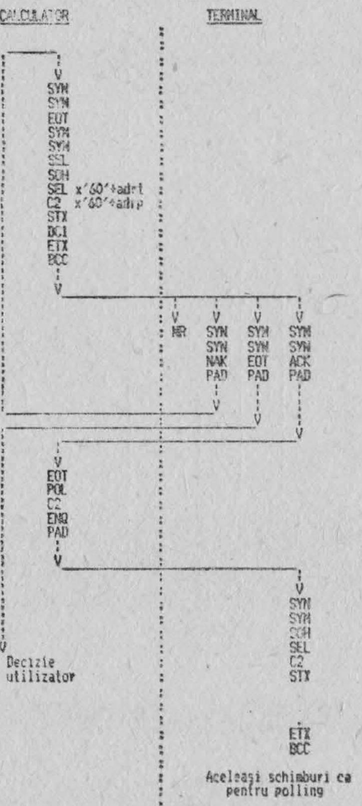


Figura 5.11. TMM-VU Mesaj de diagnostic

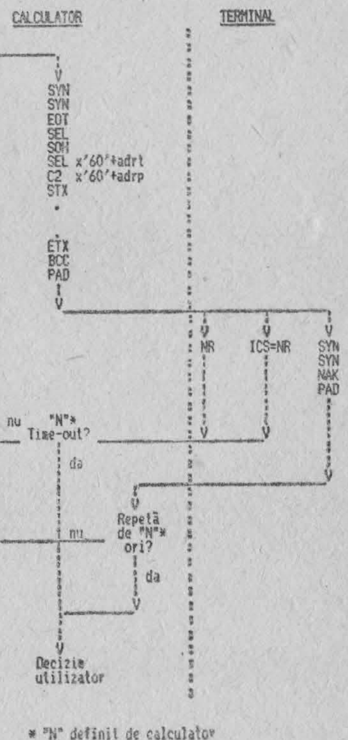


Figura 5.12. TMM-VU Recuperare erori selecţie

CALCULATOR TERMINAL

```
SYN
SYN
EOT
SYN
SYN
SEL
SOH
AD1 = SEL
AD2 x'40'+Adrp --> rezervare periferic
STX
AD3 x'50'+Adrf --> rezervare în scriere
    x'40'+Adrf --> rezervare în citire
ETX
BCC
PAD
```

NR | SYN SYN NAK PAD | SYN SYN ACK PAD

Repetă de "N"* ori? — da / nu

Revenire la TMM-VU

```
SYN
SYN
EOT
SYN
SYN
POL
ADR x'40'+Adrp --> confirmare rezervare
ENQ
PAD
```

NR | SYN SYN NAK PAD | SYN SYN ACK PAD | SYN SYN EOT PAD

Repetă de "N"* ori? — da / nu

Revenire la TMM-VU

Secvență scriere citire pe/de pe periferic

Perifericul este în curs de rezervare

* "N" este definit de calculator

Figura 5.18. TMM-VU extins. Secvența de rezervare periferic

CALCULATOR TERMINAL

```
SYN
SYN
EOT
SYN
SYN
SEL
SOH
AD1 = SEL
AD2 x'40'+Adrp
STX
AD3 x'60'+Adrf -> eliberare în citire
    x'70'+Adrf -> eliberare în scriere
ETX
BCC
PAD
```

NR | SYN SYN NAK PAD | SYN SYN ACK PAD

Repetă de "N"* ori? — da / nu

Revenire la TMM-VU

* "N" definit de calculator

Figura 5.19. TMM-VU extins. Secvența de eliberare periferic

CALCULATOR TERMINAL

SYN
SYN
EOT
SYN
SYN
POL x'40'+Adrt
ADR x'50'+Adrp
ENQ
PAD

SYN
SYN
EOT
PAD

SYN
SYN
SOH
AD1
AD2 x'40'+Adrp
STX
NOB
DC1
.
DC2
NRB
.
DC3
ETX
BCC
PAD

Tratare mesaj

S-a primit DC3?

nu

da

Eliberare periferic

Figura 5.20. TMM-VU extins. Secvența de citire de pe un periferic

CALCULATOR TERMINAL

SYN
SYN
EOT
SYN
SYN
SEL
SOH
AD1 = SEL
AD2 x'50'+Adrp
STX
NOB
DC1
.
DC2
NRB
.
DC3
BCC
PAD

SYN
SYN
NAK
PAD

SYN
SYN
ACK
PAD

SYN
SYN
EOT
PAD

Retransmite același buffer

Detectare sfîrșit fișier

nu

Transmite bufferul următor

da

Eliberare periferic

SYN
SYN
EOT
SYN
SYN
POL
ADR x'50'+Adrp
ENQ
PAD

SYN
SYN
EOT
PAD

SYN
SYN
ACK
PAD

SYN
SYN
NAK
PAD

continuă

eroare fatală

Eliberare periferic

Figura 5.21. TMM-VU extins. Secvența de scriere pe un periferic

5.3.3. DDCMP

DDCMP (Digital Data Communication Message Protocol) este un protocol general capabil să opereze în urmatoarele condiții: [2], [8], [58]

- transmisie duplex sau semi-duplex;
- linii punct la punct sau multipunct;
- comunicație serială (sincronă sau asincronă) sau paralelă,

5.3.3.1. Caracteristicile protocolului

1.Transparența informației este asigurată prin contor de caractere:

- primul caracter din mesaj este un caracter de control, identificînd tipul mesajului;
- mesajul are un antet de control, de lungime și structură fixă;
- lungimea zonei de date din mesaj este indicată în antet, deci nu este nevoie de delimitatori speciali pentru început și sfîrșit de date, deci de rezervare de coduri pentru caractere de control,

2.Controlul erorilor și recuperarea lor sînt asigurate astfel:

- pentru fiecare mesaj se include control prin CRC-16 la nivel de antet și CRC-16 pentru zona de date (dacă există);
- păstrarea secvenței mesajelor și detectarea de pierderi de mesaje se face prin numerotarea lor la emisie (modulo 256) și prin achitări incluzînd număr de mesaj;
- achitarea mesajelor primite are urmatoarele particularitați:
  - nu este necesară la nivel de mesaj; o achitare specificînd un număr de secvența N arată că toate mesajele cu număr între cel specificat la achitarea anterioara și N (modulo 256) au fost primite corect și în secvența corectă (maxim 256 de mesaje intre două achitări);
  - se poate face:
    - prin mesaj de achitare explicită cu număr de secv..ță:

      ACK-N: toate mesajele au fost primite corect, ultimul are numarul "N";

      NAK-N: mesajele pîna la numarul N au fost primite corect, iar mesajul N+1 cu eroare;
    - prin mesajele de date transmise (în antetul de mesaj existînd un cîmp pentru indicarea numărului ultimului mesaj primit corect de la partener);
  - un mesaj recepționat care nu este în secvență nu se achită; dacă pentru un mesaj "N" trimis emițătorul nu a primit răspuns după un "interval de timp de răspuns" (definit pentru implementarea respectivă), va trimite un mesaj REP-N la care partenerul va răspunde cu ACK/NAK dacă a primit/nu a primit mesajul respectiv.

3.Sînt incluse mecanisme simple pentru controlul fluxului informației:

- pentru operare semi-duplex sau multipunct, stația emițătoare poate anunța că un mesaj este ultimul emis spre partenerul curent (prin setarea unui bit "de selecție" din antetul de mesaj);
- dacă sînt disponibile, pot fi emise mai multe mesaje fără pauză între ele (în mod serial, sincron, nu mai este necesara emiterea de caractere de sincronizare între ele); acest lucru este anunțat receptorului prin bitul "de sincronizare" din antetul de mesaj.

4. Luarea de legătură inițială între doi parteneri se face printr-un schimb de mesaje START trimis de inițiatorul legăturii, STACK ("achitare start") trimis de receptorul mesajului START.

5.3.3.2. Limitările protocolului

Limitările protocolului sint:

- unele cuploare de comunicație interactioneaza cu procesorul (prin intreruperi) la nivel de mesaj și nu de caracter; în cazul in care detectarea sfirsitului de mesaj se face (hard) prin recepționarea unor anumite caractere de control, nu este posibil să se folosească protocolul DDCMP;
- nu se pot emite caractere de sincronizare dacă apar pauze între caracterele de emis; ele vor fi interpretate drept caractere utile la recepție.

5.3.3.3. Structura mesajelor de protocol

Mesajele de protocol au urmatoarele cimpuri:

CLASA (8 biti) : clasa mesajului de protocol,

- = 10000001 - mesaje de date;
- = 00000101 - mesaje de control;
- = 10010000 - mesaje de "întreținere".

CONTOR/TIP (14 biti) :

- pentru mesaje de date și de întreținere lungimea zonei de date din mesaj (în octeți);
- pentru mesaje de control:
  - = 00000001000000 ACK
  - = 00000010xxxxxx NAK - unde xxxxxx arată cauza emiterii:
    - 000001 - eroare CRC antet;
    - 000010 - eroare CRC date;
    - 000011 - răspuns la REP;
    - 001000 - buffer nedisponibil;
    - 001001 - pierdere de date la recepție ("overrun");
    - 010000 - mesaj prea lung;
    - 010001 - eroare de format în antetul receptionat;
  - = 00000001100000 REP;
  - = 00000110000000 START;
  - = 00000111000000 STACK :

BITI SEMNALIZARE (2 biti):

- bit de sincronizare;
- bit de selecție.

RASPUNS (8 biti): numărul ultimului mesaj recepționat corect.

SECVENTA (8 biti):

- pentru mesajele de date: numărul mesajului emis (modulo 256);
- pentru mesaj REP: numărul ultimului mesaj emis de stația respectivă.

ADRESA (8 biti):

- pentru operare multipunct: adresa stației secundare (atît în mesajele spre cît și în cele de la stația secundară);
- în rest nu are semnificație (se pune prin conventie valoarea 1).

CRC 1 (16 biti) : CRC-16 pentru antet (cîmpurile de mai sus).

DATE (lungime = valoare cimp CONTOR):

- pentru mesajele de date: date de trasmis;
- pentru mesajele de întreținere: folosit pentru structurarea mesajelor protocolului MOP (folosit pentru funcții de întreținere ca: încărcarea/ idarea memoriei, test linie,etc.).

CRC 2 : (16 biti) : CRC-16 pentru cimpul DATE (deci există numai la mesajele care au cimpul DATE).

5.3.3.4. Implementări existente la noi in țară

Protocolul este disponibil pe minicalculatoarele românesti din familia CORAL-INDEPENDENT și pe minicalculatoarele PDP 11/34, atît prin implementari software pentru cuploare de comunicatie sincrone (TTS 107, SI 40A) cît și la nivel hardware (cuplor MPX 40).

Protocolul DDCMP este folosit pentru comunicație între nodurile din cadrul rețelei RENOD, componentă a rețelei naționale de calculatoare.

Este în curs de realizare o implementare pentru microcalculatoarele românesti din familia M18, M118.

Momentan nu este posibilă o implementare pe calculatoare din familia FELIX C-256/512/1024, folosind cuplorul sincron CTQM, din motive de limitări hardware ale acestui cuplor (vezi punctul 3).

5.3.4. HDLC

HDCL (High-level Data Link Communication) este un protocol de tip "bit sincron" ceea ce înseamnă că: [2], [4], [59]

- este folosit pentru transmisie sincronă;
- controlul transferului informației are la bază elementul binar, ceea ce se traduce prin:
  - mesajele sînt vazute ca o secvență de biți și nu ca o secvență de caractere;
  - anumite configurații de biți, în anumite poziții din mesaj, sînt folosite pentru cadraj, definire tip mesaj, etc.;
  - nu există "caractere de control", coduri rezervate.

5.3.4.1. Cadrajul informației. Controlul stării legăturii

1.Toate mesajele încep și se termină cu o secvență de 8 biți de forma 01111110, secvență ce poartă numele de "flag". Pentru a evita apariția acestei secvențe în interiorul zonei de date a mesajului, la emisie se inserează un 0 după orice secvență de 11111 iar la recepție se înlatură acest 0 suplimentar.

2.Între două mesaje transmise se pot transmite flaguri sau o secvență de cel puțin 7 cifre 1 consecutiv; la limită, flagul de sfîrșit al unui mesaj poate servi ca flag de început pentru mesajul următor.

3.Emisia a cel puțin 7 cifre de 1 consecutiv semnifică abandonarea mesajului în curs.

4.Emisia a cel puțin 15 cifre de 1 consecutiv semnifică trecerea legăturii în "stare inactivă" (adică terminarea transmisiei de la emitător).

5.3.4.2. Caracteristicile protocolului.

1.Se consideră că dialogul de protocol are loc între două stații - una cu statut de "master", numită stație primară și una cu statut de "slave", numită stație secundară. Stația primară emite "comenzi" iar stația secundară emite "răspunsuri" la aceste comenzi; aceasta este o configuratie de legătură neechilibrată. Pentru o configurație echilibrată se consideră că la fiecare capat exista o stație "combinată" compusă dintr-o stație primară și una secundară, fiecare stație primară cooperînd cu stația secundară de la celalalt capăt al legăturii.

2.Secvențialitatea și integritatea informației transferate prin mesaje de protocol se asigură prin numerotarea mesajelor emise și achitarea numerotată a acestora de către receptor. Pentru eficiența schimbului de date, nu este necesară achitarea individuală a mesajelor. Se stabilește o "fereastră de transmisie", cu un număr maxim de mesaje ce pot fi emise fără a fi achitate, achitarea lor putîndu-se face global. Dimensiunea ferestrei este limitată de mărimea cîmpului "număr de secvență din mesaj".

5.3.4.3. Structura mesajelor

Mesajele de protocol au următoarele cîmpuri:

- FLAG: o succesiune de 8 biți.
- ADRESĂ: o succesiune de 8 biți sau, prin acord între părțile în dialog, o secvență de octeți de lungime variabilă, sfîrșitul fiind marcat printr-un octet cu primul bit pe 1, ceilalți octeți din cîmp avînd primul bit pe 0. Acest cîmp identifică stația (stațiile) secundară care este sursă/destinație a transferului respectiv.
- COMANDĂ: o succesiune de 8 sau de 16 biți ce indică un tip de comanda de la stația primară sau tipul raspunsului de la stația secundară. Acest cîmp cuprinde și numerele de secvență ale mesajelor transmise/recepționate.
- INFORMAȚIE: o succesiune de biți de date de lungime variabilă; cîmpul poate lipsi pentru anumite tipuri de mesaje.
- CRC: o succesiune de 16 biți cu rol de secvență de control al mesajului.

Observație: biții din cîmpurile ADRESĂ, COMANDĂ se transmit începînd cu bitul cel mai nesemnificativ și terminînd cu cel mai semnificativ bit.

5.3.4.4. Tipurile de mesaje

Datele propriu-zise se transferă prin mesaje de tip "informație" **I** . Supervizarea unei conexiuni se face prin mesaje de tip "supervizare" **S** si de tip "nesecvențial" **N** . Tipul unui mesaj este definit prin formatul cîmpului COMANDĂ din mesaj. Cimpul COMANDĂ poate fi o succesiune de 8 biți, caz în care are formatul dat în tabelul 5.5., sau poate fi extins pe 2 octeți, în care caz are formatul dat de tabelul 5.6.

Tabelul 5.5.

NR.BITULUI / TIP.INFORMATIEI	0	1	2	3	4	5	6	7
I	0	N(S)			P/F	N(R)		
S	1	0	S		P/F	N(R)		
N	1	1	M	M	P/F	M	M	M

Tabelul 5.6.

NR.BITULUI / TIP.INFORMATIEI	0	1	2	3	4	5	6	7	8	9 10 ...14 15
I	0	N(S)							P/F	N(R)
S	1	0	S		X				P/F	N(R)
N	1	1	M	M	U	M	M	M	P/F	X

Notațiile din tabele au urmatoarele semnificații:

- N(S): numărul (modulo 8) de secvență al mesajului emis;
- N(R): numărul (modulo 8) de secventă al următorului mesaj așteptat de la partener. Acest numar semnifică recepția corectă a mesajelor numerotate pîna la N(R)-1;
- P: cerere de răspuns imediat -în cazul unui mesaj emis de stația primară;
- F: răspuns al stației secundare la un mesaj transmis de stația primară cu P setat. Dacă stația secundară nu lucrează în mod autonom, semnifică mesaj final emis de stația secundară. De subliniat ca fiecărui P emis trebuie să-i corespunda un F recepționat și nu se poate transmite alt mesaj cu P setat pînă cînd nu s-a primit răspunsul cu F setat la P-ul anterior;

S: tipul funcției de supervizare (al comenzii de la primar și al răspunsului de la secundar):

- 00 RR (receive ready): arată că stația este pregatită pentru recepția informației sau că stația achită mesajele cu numar de secventă pîna la N(R)-1.
- 01 REJ (reject): folosit pentru a cere transmisia/retransmisia mesajelor începînd de la N(R) (implicit se confirmă cele recepționate pînă la N(R)-1 inclusiv)
- 10 RNR (receive not ready): achită mesajele recepționate pînă la N(R)-1 și arată că stația nu este (temporar) în masură să mai primească alte mesaje.
- 11 SREJ (selective reject): folosit pentru a cere transmisia/retransmisia mesajului cu numărul de secvență N(R) (implicit se achită cele pînă la N(R)-1.

X: biți rezervați cu valoare 0 dacă nu s-a definit o semnificație pentru ei.

U: bit nefolosit.

MMMMM: indica tipul comenzii "nesecvențiale" de la stația primară sau tipul răspunsului "nesecvențial" de la stația secundară:

- 11000 - comanda SARM (setare mod de răspuns autonom): în mod de lucru autonom, o stație secundară poate emite din proprie inițiativă, fără a fi invitată de stația primară. Se definește un interval de "time-out" pentru mesajele transmise; dacă acesta expiră și nu s-a primit achitarea mesajelor se reia transmisia. Pentru a evita eventualele coliziuni in cazul transmisiei bidirecționale pe circuite semi-duplex, intervalul de time-out pentru stația secundară trebuie să fie mai mare decît cel al stației primare;
- 00001 - comanda SNRM (setare mod de răspuns normal): în mod de lucru normal, stația primară controlează legatura iar stația secundară nu poate emite decît dacă primește permisiunea stației primare printr-un mesaj cu bitul P setat. După primirea permisiunii stația secundară poate emite unul sau mai multe mesaje, ultimul mesaj avînd bitul F setat;
- 11100 - comanda SABM (setare mod de lucru echilibrat asincron): folosită de stațiile combinate;
- 11011 - comanda SNRME (setare mod de lucru normal extins): folosită pentru mesajele în care cîmpul COMANDĂ este de 16 biți;

Regimurile de lucru ale imprimantei se stabilesc cu ajutorul a doi comutatori de cîte 8 poziții și un comutator de 4 poziții, care se găsesc pe panoul posterior al echipamentului, pe placa de interfață serială IS. Semnificația lor este prezentată în tabelul 4.12.

COMUTATOR	POZIȚIE	REGIMUL DE LUCRU
A1		nu se folosește
A2	ON	AUTO LINE FEED
A3	ON	8 biți pe caracter
	OFF	7 biți pe caracter
A4		nu se folosește
A5	ON	există paritate
	OFF	fără paritate
A6	ON	paritate impară
	OFF	paritate pară
A7		nu se folosește
A8	ON	2 biți de STOP
	OFF	1 bit de STOP
B1	ON	AUTO TEST
B2	ON	generator standard de caractere
	OFF	generator opțional
B3	ON	salt automat la sfîrșit pagină
B4	ON	lungime pagină de 11 inch
	OFF	lungime pagină de 12 inch
B5	ON	6 linii pe inch
	OFF	8 linii pe inch
B6	ON	132 caractere pe linie
	OFF	80 caractere pe linie
B7 B8	OFF OFF	10 cpi
	OFF ON	16.5 cpi
	ON OFF	13.6 cpi
	ON ON	10 cpi dublă densitate
C1 C2 C3	OFF OFF OFF	110 bps
	OFF OFF ON	150 bps
	OFF ON OFF	300 bps
	OFF ON ON	600 bps
	ON OFF OFF	1200 bps
	ON OFF ON	2400 bps
	ON ON OFF	4800 bps
	ON ON ON	9600 bps

10001 - răspuns CMDR (comanda recepționata corect dar rejectată). Cîmpul de informație al mesajului CMDR indică motivul rejectării și identifică mesajul rejectat, in următorul format :

1-6 : cîmpul COMANDĂ al mesajului rejectat;
9 : = 0;
10-12 : N(S) - valoarea actuală, pentru stația secundară, a contorului de emisie;
13 : = 0 ;
14-16 : N(R) - valoarea, pentru stația secundară, a contorului de recepție;
17 : = 1 pentru codul de comandă eronat;
18 : = 1 pentru neconcordanță între tipul comenzii și conținutul cîmpului INFORMAȚIE;
19 : = 1 dacă mesajul recepționat are lungime mai mare decît bufferul maxim al receptorului;
20 : = 1 dacă valoarea N(R) din mesajul recepționat este incorectă.

Pentru comenzi în format extins, cîmpurile COMANDĂ, N(S), N(R) din mesajul CMDR se lungesc corespunzator.

5.3.4.5. Implementari existente

Un subset al protocolului HDLC este folosit la nivelul "legătură de date" al interfeței de conectare a unui echipament de calcul la o rețea publică de date X.25. În acest context, sînt realizate urmatoarele implementari ale protocolului: [60]

- pe minicalculatoarele din familia CORAL - INDEPENDENT, atît integrat în software-ul pentru rețeaua de comunicație RENOD, cît și independent de aceasta (driver de comunicație plus ACP pentru protocol);
- pe microcalculatoarele din familia M-18.

Tabelul 5.4. Caracteristici generale protocoale

CARACTERISTICĂ	TMM	BSC	DDCMP	HDLC
Duplex	NU (DA în TMM-UC)	NU	DA	DA
Semiduplex	DA (NU în TMM-UC)	DA	DA	DA
Serial	DA	DA	DA	DA
Paralel	NU	NU	DA	NU
Transparență date	char. stuf-fing	char. stuf-fing	contor	bit stuf-fing
Asincron	NU	NU	DA	NU
Sincron	DA	DA	DA	DA
Punct la punct	DA	DA	DA	DA
Multipunct	DA	DA	DA	DA
Detectare erori	LRC par VRC impar	CBC 16	CRC 16	CRC CCITT
Retransmisie pentru recuperare erori	DA	DA	DA	DA

5.4. DESCRIERE PROTOCOALE ASINCRONE

Protocoalele asincrone disponibile pe diversele tipuri de terminale permit operații ce merg de la transferul caracter cu caracter semi-duplex (compatibile teletype) pina la operarea in mod bloc. [52],[53] Cele mai cunoscute protocoale asincrone pot fi împarțite în trei mari categorii:

A. Standard
Compatibile teletype sau utilizînd metode de "hand shake" (comunicare cu confirmare prin folosirea caracterelor DC1, DC2).

B. Canal principal
Protocolul utilizează caractere speciale pentru delimitarea și controlul comutării liniei.

C. Canal secundar
Utilizează canalul secundar (pentru modemuri care au această facilitate) pentru a semnala comutarea liniei.

A. Operarea la viteze mari pentru protocol compatibil teletype.

Daca debitul de caractere trimise la terminal depașește viteza de prelucrare internă a terminalului pot apare pierderi de caractere (acest lucru apare în mod obișnuit la viteze mai mari de 4800 bps). Simptomul acestei probleme se manifestă in funcție de tipul terminalului (desincronizari la terminalele KSR sau receive-only, apariția unor caractere speciale la terminalele de tip display).

Exista trei metode (principale) pentru eliminarea acestui inconvenient:

1. Utilizarea unei proceduri de tip întrebare-răspuns între terminal și calculator prin care calculatorul așteaptă prelucrarea textului transmis și confirmarea acestei prelucrări de la terminal (ex: pentru HP 2645A, B, C). Calculatorul trimite ENQ după fiecare 80 caractere și așteaptă ACK de la terminal pentru transmisie.
2. Utilizarea unei metode de temporizare prin inserarea, în software-ul de aplicație sau de sistem, a unor caractere în transferul de date de la calculator la terminal. Se utilizează de obicei caracterul NUL. Fiecare caracter NUL are efectul a 4 milisecunde întîrziere la 2400 bps și 2 milisecunde la 4800 bps.

Pentru a calcula timpul de întîrziere este necesară existența unei liste cu timpi de prelucrare a diverselor funcții de terminal furnizate, de obicei, de firmă.

3. Utilizarea mecanismului X-ON/X-OFF (DC1/DC3).

Este metoda cea mai răspindită de temporizare a transmisiei dinspre calculator. În momentul în care terminalul detectează un procent de umplere al bufferului de recepţie trimite spre calculator caracterul DC3 (X-OFF) prin care anunţă că nu mai poate primi caractere. După prelucrarea caracterelor primite (sau a unui procent din acestea) se transmite spre calculator caracterul DC1 (X-ON) prin care se anunţă reluarea transmisiei.

Acest mod de lucru este implementat la cvasitotalitatea terminalelor asincrone existente în economie.

B. Protocolul de canal principal utilizează caractere de control pentru cadrajul fiecărei transmisii de date. Aceste caractere de delimitare indică staţiei receptoare că transmisia a început sau s-a sfîrşit, şi pot conţine şi caractere de adresare (specifice operaţiilor de polling/selecţie). Terminalele sînt interogate individual, atît pentru transmisie cît şi pentru recepţie. Terminalele răspund fie cu mesaj de date (dacă există) fie cu masaj de stare (pentru identificarea stării curente a terminalului).

Protocoalele tipice acestei categorii sînt cele întîlnite la terminalele IBM 2848, HP 2645, ELITE 2500.

5.5. TEHNICI DE IMPLEMENTARE PROTOCOALE

Protocoalele de teletransmisie pot fi implementate prin:

- software;
- firmware;
- hardware;
- mixt.

Iniţial, protocoalele erau implementate prin software, dar treptat, pentru descongestionarea calculatorului, o serie de controale au fost preluate de cuploarele de teletransmisie.

Actualmente, cuplorul tinde a deveni un procesor capabil de a lucra în mai multe protocoale.

Scopul implementarii unui protocol sincron este acela de a degreva programatorul de aplicaţie de greaua muncă pe care o reprezinta programarea operaţiilor fizice şi procedurilor de teletransmisie, lăsînd la discreţia sa numai descrierea reţelei şi problemele de logică. Apariţia unor noi terminale sau echipamente de teleprelucrare ce posedă protocoale sincrone diferite de cele existente pe calculatorul gazdă impune introducerea acestor protocoale în metoda de acces generală a acestuia sau realizarea de aplicaţii specifice bazate pe integrarea protocolului într-un modul de acces (proces, task, AST).

Tehnicile de implementare depind de calculatorul gazda dar, în general, se bazează pe facilitatea de "întrerupere adiţională" (FELIX), AST (mini), întrerupere (micro, terminale).

5.5.1. Tehnici pe calculatoare FELIX C

Derularea unei I/E tipice pentru un protocol sincron pe FELIX este dată în figura 5.22. [37], [47], [63]

Se observă că execuţia unei macroinstrucţiuni (MI) se traduce printr-un salt în modulul de analiză şi control care verifică coerenţa apelurilor şi lansează secvenţa de program asociată tranziţiei de protocol cerută de programul de aplicaţie. Această tranziţie corespunde unei faze din derularea protocolului asociat unei MI.

Prima fază, denumită şi faza de iniţializare (F0), efectuează următoarele controale asupra apelului de MI:

- tipul de SEND/RECEIVE este utilizabil cu echipamentul descris
- tipul de SEND/RECEIVE este compatibil cu starea logică a liniei
- linia este afectată partitiei
- doua MI de tip SEND/RECEIVE sînt separate de o MI de asteptare

Dacă toate aceste controale sînt pozitive, controlul este trecut secvenţei de tratare a fazei 1 corespunzătoare tipului de macroinstrucţiune programată care se termină printr-un EXUP. Supervizorul pune bitul eveniment al CB-ului asociat pe "1". Apariţia unei întreruperi relansează supervizorul care resetează bitul eveniment şi trece imediat controlul către rutina aditionala de întreruperi a carei adresă se află în CB (extins la 4 cuvinte). Principial, aceasta corespunde tratării unei secvenţe de protocol asociată fazei N şi este compusă din următoarele secvenţe de program:

- determinare răspunsuri cuplor SHT (SIO, HIO, TIO), AIO şi TDV care, de obicei, este plasată intr-un modul general ce întoarce un index de ramificare intr-o tabelă de tratare a cazurilor de mai jos;
- tratare I/E corectă d.p.d.v. al transmisiei; în această secvenţă se verifică practic coerenţa protocolului;
- tratare erori de transmisie datorate echipamentelor de teletransmisie (modemuri, cuploare, linii);
- tratare informaţie invalidată de emiţător;
- tratare apariţie time-out hardware (T01 utilizat în sincron);
- tratare recepţie EOT.

Daca faza N detectează necesitatea relansării unei I/E (determinată de succesiunea tranziţiilor protocolului sau mecanismele de tratare erori) atunci bitul eveniment va fi repus de supervizor pe "1" (operaţia EXUP) şi programul utilizator nu va fi relansat (fiind în aşteptare pe acest bit eveniment). Dacă faza N determină sfîrşitul secvenţei de protocol în condiţii corecte sau prin epuizarea mecanismelor de tratare erori, atunci lasă bitul eveniment resetat şi programul utilizator va fi startat după MI de aşteptare sfîrşit I/E cu anunţarea acestuia asupra condiţiilor de terminare I/E.

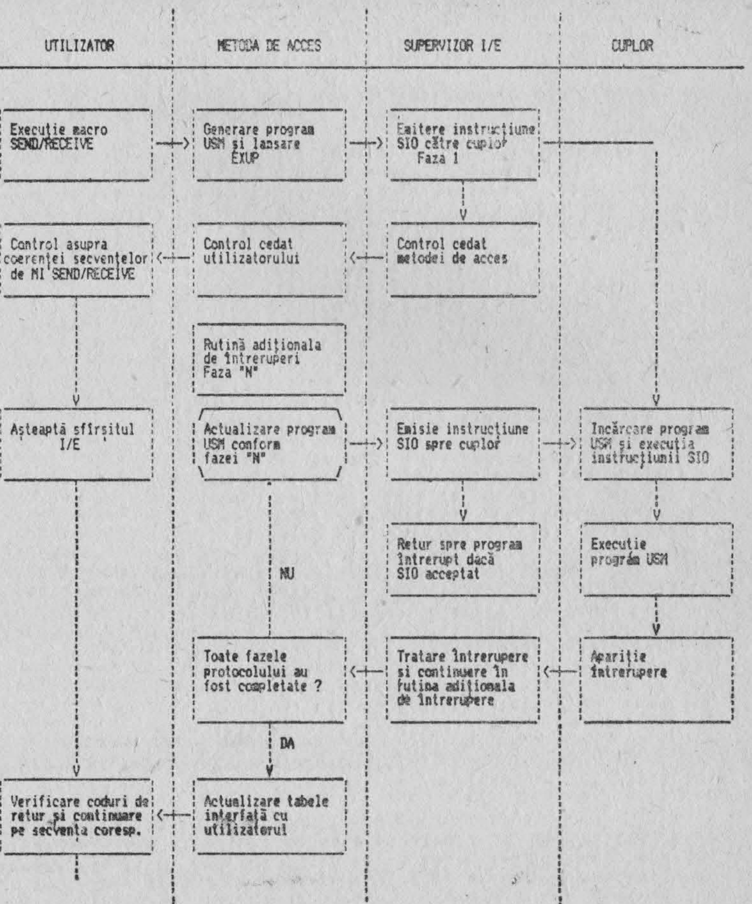


Figura 5.22. Secvenţă tipică de I/E pentru protocol sincron pe FELIX

5.5.2. Tehnici pe minicalculatoare

Implementarea unui protocol pe un calculator trebuie să aibă în vedere:

- realizarea tuturor funcţiilor protocolului (cadraj, controlul erorilor, al fluxului de informaţie, etc);
- utilizarea optimă a resurselor hardware şi a facilităţilor sistemului de operare;
- asigurarea unei interfeţe cît mai simple cu utilizatorul protocolului (în general, un program de aplicaţie).

În implementarea pe minicalculator sînt implicate de regulă următoarele componente:

- cuploare de comunicaţie;
- drivere aferente acestora;
- programe (de tip 'task utilizator' sau ACP) specifice protocolului ('programe de control pentru protocol').

Rezolvarea problemelor de implementare poate fi împărţită în diferite moduri între aceste componente:

A. sa se facă integral la nivel de cuplor (exemplu: cuplorul DMC care implementează protocolul DDCMP);

B. cuplorul să realizeze cadrajul: să se sincronizeze pe început de mesaj - în cazul transmisiei sincrone , să detecteze caracterul de început de mesaj şi sfîrşitul de mesaj - prin declanşare de întrerupere - daca este cuplat prin DMA (exemplu: cuplorul OMM-H pentru procedura hardware a Organizatiei Mondiale de Meteorologie) sau prin setarea unui bit în registrul de periferic - dacă lucrează prin întrerupere la fiecare caracter (de exemplu: cuploarele SI-40 de la CORAL şi TTS-107 de la INDEPENDENT, în mod de lucru sincron orientat caracter); cuplorul mai poate face şi control/generare de paritate la nivel de caracter sau de bloc;

C. cuplorul realizează numai transferul de caractere şi -dacă e cazul- sincronizarea; cadrajul se face la nivel de driver; controlul de paritate se poate face în acest caz la nivelul driverului sau la nivelul programului de control pentru protocol (exemplu: cuploarele SI-40 şi TTS-107 pot fi folosite in acest mod).

În cazurile B si C, este problematică posibilitatea de a realiza integral la nivel driver partea de control a protocolului, ţinînd cont de faptul că un driver în sistemele de operare RSX 11-M, AMS, MINOS lucrează la nivel 'fork' sau de întrerupere,are structuri de date şi structuri de rutine care trebuie definite într-un anumit mod [49], [61].

Există mai multe posibilităţi de implementare a nivelului 'program de control pentru protocol':

- prin integrarea într-o aplicaţie specializată a protocolului, ca modul al acesteia (de exemplu: emulatorul de staţie ARIEL - EMMA);
- prin realizarea unor 'module de acces' (module-obiect) care se leagă la fiecare aplicaţie care utilizează protocolul, utilizatorului fiindu-i oferite macrouri pentru apelarea simplă a funcţiilor de protocol - modalitate analoagă cu realizarea SGT din sistemul SIRIS 3 (de exemplu: produsul TELEMINI);
- prin realizarea unor ACP-uri asociate driverului; în acest caz, interfaţa utilizatorului cu protocolul se realizează la nivel de directivă QIO; în acest fel, protocolul poate fi utilizat în paralel şi în mod independent de către mai mulţi utilizatori (de exemplu: implementarea standardului X25 - varianta ne-integrată în sotfware-ul de reţea);
- prin utilizarea unor zone de memorie comună (COMMON) între un task de control al protocolului şi taskurile care utilizează protocolul, cu folosirea unor mecanisme de sincronizare şi protecţie pentru accesul la zonele comune (exemplu: sistemul tranzacţional de gestiune a reţelei de terminale - protocol BSC).

La fiecare nivel, trebuie avut în vedere ca implementarea să 'ţină pasul' cu comunicaţia pe linie: ţinînd cont de viteza maxima de transmisie pe linie, de convenţiile de 'time-out' şi reluări din protocol, să nu poata apare pierderi de caractere sau de mesaje, desincronizări în protocol. Pentru aceasta:

- pentru cuploare care întrerup la nivel de caracter, rutinele de întrerupere din drivere să nu fie prea lungi;
- e preferabil ca taskurile care implementează protocolul să nu fie checkpointabile şi să trateze cu prioritate şi cît mai rapid mesajele recepţionate de pe linie (folosind rutine AST, de exemplu).

6. CODURI

6.1. GENERALITĂȚI

În comunicație toate informațiile trebuie puse sub formă de simboli. Semnificația precisă a acestor simboli este evident fundamentală dar este o simplă convenție între emițătorul mesajului și destinatar. [3]

Codul este regula de corespondență, între informație și configurația binară asociată, fiecare informație corespunzînd în general unei singure configurații binare. [3].

Lungimea codului depinde de numărul de simboli **N** care trebuie reprezentați. Cu 2 elemente binare se obțin 4 combinații, deci se pot codifica 4 simboli. Generalizînd, cu **n** elemente binare se pot reprezenta 2^n simboli.

Pentru a codifica **N** simboli cu $2^{(n-1)} < N < 2^n$, trebuie deci cel puțin **n** biți. Toate codurile uzuale utilizează valori de **n** cuprinse între 5 și 8 biți.

Informațiile care se transmit, deci care se codifică, sînt un ansamblu compus din următoarele elemente, numite caractere:

- 10 cifre din sistemul de numerotare zecimal uzual;
- 26 litere ale alfabetului (52 dacă se utilizează majuscule și minuscule);
- un număr de semne de punctuație și operatori aritmetici (ex: +, -, ?, /, $ etc.);
- un ansamblu de caractere de comandă destinate fie pentru a ușura transmisia informației, fie pentru a comanda oprirea sau pornirea de la distanță a perifericelor, fie pentru paginarea la nivelul receptorului.

Este de dorit ca echipamentele care utilizează coduri diferite să poată comunica. Pentru aceasta, este necesară o conversie de cod care deobicei se realizează de către unul din "partenerii" comunicației (în general emitorul).

6.2. TABELE DE CODURI

6.2.1. Codul Baudot

Codul Baudot (cod telegrafic cu 5 biți sau Alfabetul Internațional nr.2 sau codul CCITT nr.2) este descris în tabelul 6.1.

Este un cod folosit în rețeaua telegrafică și are 5 biți (deci nu ar permite decit codificarea a 32 de caractere, insuficient pentru litere și pentru cifre). Pentru codificarea celor 58 de caractere se folosesc secvențele de comandă "inversiune litere" și "inversiune cifre" pentru a desemna setul de caractere cu care se lucrează.

6.2.2. Codul ASCII

Codul ASCII (American Standard Code for Information Interchange), cunoscut și sub numele de ISO 2022 cu 7 biți sau alfabet internațional nr. 5 sau cod CCITT nr.5, este un cod pe 7 biți completat eventual de un bit de paritate și permite codificarea a 128 de caractere. Codul prevede și integrarea unor caractere specific naționale.

Codul ASCII (standard CCITT) este descris în tabelul 6.2. iar codul cu 7 biți standard pentru R.S.România, in tabelul 6.3. (STAS 9369-76).

6.2.3. Codul EBCDIC

Codul EBCDIC (Extended Binary Coded Decimal Interchange Code) este un cod pe 8 biți fără bit de paritate, introdus de firma IBM.

Codul EBCDIC este descris în tabelul 6.4.

Bits 7,6,5		000	001	010	011	100	101	110	111
Bits 4321	Hex	0	1	2	3	4	5	6	7
0000	0	NUL	DLE	SP	0	@	P	`	p
0001	1	SOH	DC1	!	1	A	Q	a	q
0010	2	STX	DC2	"	2	B	R	b	r
0011	3	ETX	DC3	#	3	C	S	c	s
0100	4	EOT	DC4	$	4	D	T	d	t
0101	5	ENQ	NAK	%	5	E	U	e	u
0110	6	ACK	SYN	&	6	F	V	f	v
0111	7	BEL	ETB	'	7	G	W	g	w
1000	8	BS	CAN	(	8	H	X	h	x
1001	9	HT	EM	)	9	I	Y	i	y
1010	A	LF	SUB	*	:	J	Z	j	z
1011	B	VT	ESC	+	;	K	[	k	{
1100	C	FF	FS	,	<	L	\	l	|
1101	D	CR	GS	-	=	M	]	m	}
1110	E	SO	RS	.	>	N	^	n	~
1111	F	SI	US	/	?	O	_	o	DEL

Tabelul 6.2. Codul ASCII (ISO 2022)

	LITERE	CIFRE	COD		LITERE	CIFRE	COD
1	A	–	11000	23	W	2	11001
2	B	?	10011	24	X	/	10111
3	C	:	01110	25	Y	6	10101
4	D	+	10010	26	Z	+	10001
5	E	3	10000	27	CR		00010
6	F	E'	10110	28	LF		01000
7	G	%	01011	29	litere		11111
8	H	H	00101	30	cifre		11011
9	I	8	01100	31	spațiu		00100
10	J	Bell	11010	32	neutilizat		00000
11	K	(	11110				
12	L	)	01001				
13	M	.	00111				
14	N	,	00110				
15	O	9	00011				
16	P	0	01101				
17	Q	1	11101				
18	R	4	01010				
19	S	'	10100				
20	T	5	00001				
21	U	7	11100				
22	V	=	01111				

Tabelul 6.1. Codul Baudot

Bits 0,1		00				01				10				11			
Bits 2,3		00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
Bits 4567	Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE			SP	&	-						{	}	\	0
0001	1	SOH	DC1 XON					/		a	j	"		A	J		1
0010	2	STX	DC2		SYN					b	k	s		B	K	S	2
0011	3	ETX	DC3 XOFF							c	l	t		C	L	T	3
0100	4									d	m	u		D	M	U	4
0101	5	HT		LF						e	n	v		E	N	V	5
0110	6		BS	ETB						f	o	w		F	O	W	6
0111	7	DEL		ESC	EOT					g	p	x		G	P	X	7
1000	8		CAN							h	q	y		H	Q	Y	8
1001	9		EM						`	i	r	z		I	R	Z	9
1010	A					[	]	¦	:								
1011	B	VT				.	$	,	#								
1100	C	FF	FS		DC4	<	*	%	@								
1101	D	CR	GS	ENQ	NAK	(	)	_	'								
1110	E	SO	RS	ACK		+	;	>	=								
1110	F	SI	US	BEL	SUB	!	^	?	"								

Figura 6.4. Codul EBCDIC

Bits 7,6,5		000	001	010	011	100	101	110	111
Bits 4321	Hex	0	1	2	3	4	5	6	7
0000	0	NUL	DLE	SP	0	Ţ	P	ţ	p
0001	1	SOH	DC1	!	1	A	Q	a	q
0010	2	STX	DC2	"	2	B	R	b	r
0011	3	ETX	DC3	#	3	C	S	c	s
0100	4	EOT	DC4	$	4	D	T	d	t
0101	5	ENQ	NAK	%	5	E	U	e	u
0110	6	ACK	SYN	&	6	F	V	f	v
0111	7	BEL	ETB	'	7	G	W	g	w
1000	8	BS	CAN	(	8	H	X	h	x
1001	9	HT	EM	)	9	I	Y	i	y
1010	A	LF	SUB	*	:	J	Z	j	z
1011	B	VT	ESC	+	;	K	Ă	k	ă
1100	C	FF	FS	,	<	L	Â	l	â
1101	D	CR	GS	-	=	M	Î	m	î
1110	E	SO	RS	.	>	N	Ş	n	ş
1111	F	SI	US	/	?	O	_	o	DEL

Tabelul 6.3. Codul cu 7 biţi standard R.S.R. (STAS 9369-76)

Tabelul de corespondență coduri ASCII - EBCDIC

ASCII		EBCDIC	SIMBOL	ASCII		EBCDIC	SIMBOL
OCTAL	HEXA	HEXA		OCTAL	HEXA	HEXA	
00	00	00	NUL	100	40	7C	@
01	01	01	SOH	101	41	C1	A
02	02	02	STX	102	42	C2	B
03	03	03	ETX	103	43	C3	C
04	04	37	EOT	104	44	C4	D
05	05	2D	ENQ	105	45	C5	E
06	06	2E	ACK	106	46	C6	F
07	07	2F	BEL	107	47	C7	G
10	08	16	BS	110	48	C8	H
11	09	05	HT	111	49	C9	I
12	0A	25	LF	112	4A	D1	J
13	0B	0B	VT	113	4B	D2	K
14	0C	0C	FF	114	4C	D3	L
15	0D	0D	CR	115	4D	D4	M
16	0E	0E	SO	116	4E	D5	N
17	0F	0F	SI	117	4F	D6	O
20	10	10	DLE	120	50	D7	P
21	11	11	DC1	121	51	D8	Q
22	12	12	DC2	122	52	D9	R
23	13	13	DC3	123	53	E2	S
24	14	3C	DC4	124	54	E3	T
25	15	3D	NAK	125	55	E4	U
26	16	32	SYN	126	56	E5	V
27	17	26	ETB	127	57	E6	W
30	18	18	CAN	130	58	E7	X
31	19	19	EM	131	59	E8	Y
32	1A	3F	SUB	132	5A	E9	Z
33	1B	27	ESC	133	5B	4A	[
34	1C	1C	FS	134	5C	E0	\
35	1D	1D	GS	135	5D	5A	]
36	1E	1E	RS	136	5E	5F	^
37	1F	1F	US	137	5F	6D	
40	20	40	SP	140	60	79	`
41	21	4F	!	141	61	81	a
42	22	7F	"	142	62	82	b
43	23	7B	#	143	63	83	c
44	24	5B	$	144	64	84	d
45	25	6C	%	145	65	85	e
46	26	50	&	146	66	86	f
47	27	7D	'	147	67	87	g
50	28	4D	(	150	68	88	h
51	29	5D	)	151	69	89	i
52	2A	5C	*	152	6A	91	j
53	2B	4E	+	153	6B	92	k
54	2C	6B	,	154	6C	93	l
55	2D	60	-	155	6D	94	m
56	2E	4B	.	156	6E	95	n
57	2F	61	/	157	6F	96	o
60	30	F0	0	160	70	97	p
61	31	F1	1	161	71	98	q
62	32	F2	2	162	72	99	r
63	33	F3	3	163	73	A2	s
64	34	F4	4	164	74	A3	t
65	35	F5	5	165	75	A4	u
66	36	F6	6	166	76	A5	v
67	37	F7	7	167	77	A6	w
70	38	F8	8	170	78	A7	x
71	39	F9	9	171	79	A8	y
72	3A	7A	:	172	7A	A9	z
73	3B	5E	;	173	7B	C0	{
74	3C	4C	<	174	7C	6A	¦
75	3D	7E	=	175	7D	D0	}
76	3E	6E	>	176	7E	A0	~
77	3F	6F	?	177	7F	07	DEL

6.3. CARACTERE FUNCTIONALE

Comunicația de date necesită și caractere funcționale (comenzi) care facilitează transmisia de date. Acestea au făcut obiectul unor standardizări la nivel ISO, fiind utilizate cu aceeași semnificație în toate protocoalele sincrone orientate caracter. Există însă protocoale care pe lîngă caracterele normalizate utilizeaza și secvențe specifice (vezi capitolul 5, protocolul BSC). Lista caracterelor normalizate utilizate în teletransmisie este: **ACK, DLE, ENQ, EOT, ETB, ETX, NAK, PAD, SOH, STX.**

Descrierea semnificației caracterelor funcționale este prezentată în tabelul 6.5.

În tabela alăturală sînt date corespondențele de coduri ASCII-EBCDIC.

COMANDA	NUME	DESCRIERE	TIP
ACK	ACKnowledge	Achitare pozitivă	TC
BEL	BELl	Semnal acustic	
BS	Back Space	Revenire cursor pe poziția anterioară	FE
CAN	CANcel	Anulare	
CR	Carriage Return	Retur la început de rînd	FE
DC1	Device Control 1	Comenzi auxiliare	
DC2	Device Control 2	folosite specific	
DC3	Device Control 3	de fiecare	
DC4	Device Control 4	protocol	
DEL	DELete	Șterge caracter	TC
DLE	Data Link Escape	Specific protocol	
EM	End of Medium	Sfîrșit suport	
ENQ	ENQuiry	Interogare	TC
EOT	End Of Transmission	Sfîrșit comunicație (transfer)	TC
ESC	ESCape	Debut secvență comandă	
ETB	End of Transmission Block	Sfîrșit bloc de transmisie	TC
ETX	End of TeXt	Sfîrșit de text	TC
FF	Form Feed	Salt de pagină	FE
FS	File Separator	Separator fișier	IS
GS	Groupe Separator	Separator grup	IS
HT	Horizontal Tabulation	Tabulare orizontală	FE
LF	Line Feed	Salt la linie	FE
NAK	Negativ AcKnowledge	Achitare negativă	TC
NL	New Line	Linie nouă	
NUL	NULl	Caracter nul	
RS	Record Separator	Separator articol	IS
SI	Shift-In	Început secvența comandă	
SO	Shift-Out	Sfîrșit secvență comandă	
SOH	Start Of Heading	Început antet	TC
SP	SPace	Spațiu	
STX	Start of TeXt	Început text	TC
SUB	SUBstitution	Înlocuire	
SYN	SYNchronous idle	Caracter de sincronizare	TC
US	Unit Separator	Separator de unitate articol	IS
VT	Vertical Tabulation	Tabulare verticală	FE
FE	Format Effector	Comenzi de punere în pagină	
IS	Information Separator	Comenzi de separare informație	
TC	Transmission Control	Comenzi de control transmisie	

Tabelul 6.5. Caractere funcționale

7. CONTROLUL ERORILOR

În orice transmisie de date la distanță este posibilă apariția unor erori datorate fie imperfecțiunilor circuitelor utilizate, fie unor zgomote apărute aleator. Din această cauză este necesară utilizarea unor metode de detectare a unor erori și - eventual - de corectare a lor. [2], [8].

7.1. DETECTARE ERORI

O eroare pe linie se traduce prin interpretarea greșită a unor biți (1 drept 0 sau invers). Detectarea și corectarea erorilor sînt - de obicei - operații separate. De cele mai multe ori detectarea se face prin metode hardware iar corectarea prin metode software.

Metodele de detectare a erorilor variază în funcție de modul de transmisie, de codul și de protocolul utilizat. Majoritatea metodelor se bazează pe introducerea unor informații suplimentare (redundante) în șirul de biți transmis.

O metodă foarte simplă de detectare și corectare este cea a transmisiei în ecou. În acest caz toate datele transmise de un canal sînt retransmise de receptor emitentului. Metoda este deficitară din două motive: viteza și necesitatea utilizării unor linii duplex. Ea este utilizată doar în situațiile cînd viteza nu este un factor critic și la distanțe mici (cînd costul unei linii duplex nu este prohibitiv). Este mult utilizată pentru terminale de tip consolă la care controlul transmisiei se poate face vizual și în cazul conexiunilor directe. Posibilitatea nedetectării unei erori este foarte mică, necesitînd alterarea aceluiași bit și la emisie și la recepție.

O alta metodă de detectare a erorilor este introducerea unui bit suplimentar (numit bit de paritate sau VRC -vertical redundancy checking -) la fiecare caracter transmis. În funcție de tipul controlului de paritate utilizat (par sau impar), acest bit ia valoarea 0 sau 1 atunci cînd numărul biților de 1 din caracter este par, respectiv impar. Bitul de paritate este transmis ca cel mai semnificativ bit din caracter.

Ex:

PARITATE PARĂ		PARITATE IMPARĂ	
P		P	
1	0 1 1 0 0 0 1	0	0 0 1 1 0 0 1
0	1 0 0 0 0 1 0	1	1 0 0 0 0 1 0

p=(suma biți)modulo 2   p=(suma biți)modulo 2 + 1

Din motive electronice, la transmisiile sincrone se folosește doar paritatea impară (pentru a apare cel puțin o tranziție pe caracter).

Metoda introduce relativ puțină informație redundantă. În plus, circuitele (sau secvențele de program) introduse atît pentru calcul în transmisie cît și pentru control în recepție sînt puțin costisitoare.

Se observă ușor că apariția unui număr par de biți eronați într-un caracter nu poate fi detectată. Din această cauză, bitul de paritate, ca singurul control al corectitudinii, se folosește în transmisii la distanțe mici și, în general, în mod de lucru asincron.

Pentru micșorarea probabilității nedetectării unor erori, la sfîrșitul blocului de date se adaugă unul sau mai multe caractere de control numite BCC (Block Control Character). Aceste caractere sînt uzuale în modurile de transmisie sincrone. BCC-urile sînt de două tipuri:

- LRC (Longitudinal Redundancy Check) format dintr-un singur caracter;
- CRC (Cyclic Redundancy Check) format din două sau mai multe caractere.

În transmisia sincronă, atunci cînd se utilizează paritatea ca metodă de detecție a erorii, la sfîrșitul fiecărui bloc este introdus un caracter LRC (Longitudinal Redundancy Checking) al cărui biți reprezintă paritatea pară longitudinală a blocului.

Ex:

VRC	CARACTERE (DATE)	
0	0 1 1 0 0 0 1	
1	1 0 0 0 0 1 0	
0	1 1 0 0 1 0 0	
0	0 1 0 0 0 0 0	
1	1 0 0 0 0 0 1	
0	1 1 0 0 0 0 1	
0	0 0 1 0 1 1 1	LRC

Cel mai simplu caz de nedetectare a erorii implica aparitia a 4 biti eronati asezati (doi cite doi) pe aceeasi orizontala si verticala.

LRC-ul se transmite dupa ultimul caracter util din bloc (dupa un caracter terminator de bloc).

In mod uzual, VRC-ul si LRC-ul se folosesc in transmisiile in cod ASCII (de 7 biti utili).

O alta metoda de control al informatiei transmise este utilizarea unui CRC (Cyclic Redundancy Checking). Metoda este foarte sigura (la lungimi rezonabile de blocuri de date), avind s avantajul introducerii unei redundante mici. Dezavantajul p incipal este costul relativ ridicat - comparativ cu celelalte metode - al echipamentului sau secventelor de program suplimentare.

CRC-ul este introdus la sfirsitul blocului de date (ca si LRC-ul). Indiferent de numarul caracterelor continute de bloc, lungimea sa este constanta (doua caractere).

Principial, CRC-ul este restul impartirii a doua polinoame binare. Deimpartitul este reprezentat de sirul de biti continut de blocul de date, iar impartitorul de o secventa de biti fixa (numita polinom de generare).

Un şir de "n" biţi (notaţi "n-1" pîna la 0) este reprezentat polinomial astfel:

$$P(X)=C_{n-1}X^{n-1}+C_{n-2}X^{n-2}+\ldots+C_1X^1+C_0$$

unde C_i,

poate fi 0 sau 1 în funcţie de valoarea bitului din poziţia "i".

Puterea polinomului corespunde numarului de biţi din bloc.

Operatorul de adunare este de fapt operatorul SAU-EXCLUSIV. Cu acelaşi operator se lucrează şi în timpul împărţirii.

CRC-ul, în reprezentare polinomială, satisface relaţia:

$$X^nP(X)+R(X)=C(X)+G(X)$$

unde:

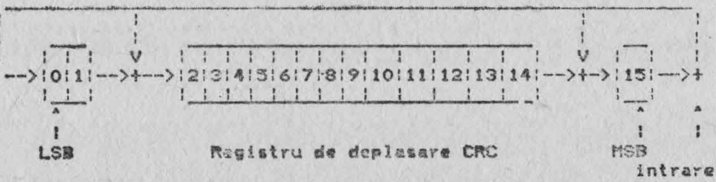
- "n" este puterea polinomului generator $G(x)$;
- $P(x)$ reprezintă şirul datelor transmise;
- $R(x)$ este CRC-ul;
- $C(x)$ este cîtul împărţirii dintre:

$$X^nP(X) \text{ şi } G(X).$$

Şirul de biţi transmişi este:

$$T(X)=X^nP(X)+R(X)$$

CRC-ul este calculat hardware cu ajutorul unui registru de deplasare în care, în timpul deplasării, sînt introduse reacţii de la bitul cel mai semnificativ.



Operaţia "+" reprezintă SAU-EXCLUSIV.

La fiecare intrare a unui bit se execută o deplasare spre dreapta (de la LSB spre MSB).

Iniţial (la începutul transmisiei) registrul de CRC este încărcat cu o valoare (de obicei 0). La sfîrşitul recepţiei, după ce şi CRC-ul primit este introdus în registru, valoarea registrului de acumulare trebuie să aibă o valoare predefinită (uzual 0). În caz contrar mesajul primit este eronat.

Polinoamele de generare cele mai uzuale sînt:

$X^{12}+X^{11}+X^3+X^2+X+1$
pentru CRC-12 (cod de 6 biţi),

$X^{16}+X^{11}+X^5+1$
pentru CRC-CCITT,

$X^{16}+X^{15}+X^2+1$
pentru CRC-16.

Transmisia cu BCC de tip CRC este utilizată la:

- transmisii utilizînd coduri de opt biţi (ex. EBCDIC în BSC şi TMM-UC);
- protocoale transparente orientate caracter (ex. DDCMP);
- protocoale de tip bit-sincron (ex. SDLC,HDLC).

În timp ce realizarea hard a acestui tip de control nu aduce după sine întîrzieri în transmisie, metodele soft sînt fie lente, dar economice în memorie, fie acceptabil de rapide, dar cu penalizări in spaţiul de memorie afectat programului (datorita necesităţii unor tabele de corecţie).

7.2. CORECTARE ERORI

Detecţia erorilor este doar primul pas în validarea unei transmisii corecte. Odată sesizată o eroare, receptorul trebuie să semnaleze emitătorului eroarea pentru a fi corectată. Procedurile de tratare erori depind de modul de transmisie, de tipul liniei şi de protocolul utilizat. [3]

În transmisia cu ecou, utilizată mai ales în conexiunile dintre un calculator şi o consolă, corecţia se face manual, de către operator.

În cazul transmisiilor sincrone sau asincrone cu protocol, la detectarea unei erori se face şi cerere de retransmisie a ultimului sau ultimelor blocuri. Această metodă se numeşte transmisie cu ARQ (Automatic ReQuest). ARQ este de două tipuri:

- cu aşteptare;
- continuu.

La ARQ cu așteptare, după fiecare transmisie de bloc de date, emițătorul așteaptă o confirmare din partea receptorului. În cazul în care răspunsul nu vine într-o perioadă predefinită, sînt declanșate proceduri de tratare time-out pe linie.

O metodă ARQ cu așteptare este specifică protocoalelor adaptate liniilor semi-duplex (la care se poate transmite într-un singur sens la un moment dat); exemplu: TMM-VU. Pentru a se mări fiabilitatea transmisiei, se utilizează metode prin care să se evite dublarea unor blocuri. De exemplu, la BSC există două caractere de confirmare (ACK0 și ACK1) pentru confirmarea blocurilor pare, respectiv impare. La TMM-RB, fiecărui bloc îi este asociat un număr (NOB), utilizat la confirmare.

Deși lentă, metoda dă bune rezultate pe liniile de proastă calitate. În plus, ea este economicoasă în utilizarea memoriei, deoarece emițătorul trebuie să păstreze - pentru retransmisie - doar ultimul bloc de date.

ARQ continuu este utilizat mai ales în protocoale moderne (DDCMP, HDLC) care pot fi adaptate liniilor duplex sau, eventual, liniilor semi-duplex de bună calitate (cu puține erori și permițînd utilizarea unui al 2-lea canal de mică viteză). La această metodă, emițătorul nu așteaptă confirmarea la fiecare bloc, aceasta fiind trimisă la un numar de blocuri. O confirmare pe bloc semnifică acceptarea tuturor blocurilor transmise de la ultima confirmare. La toate protocoalele ce pot fi folosite cu ARQ continuu, blocurile de date trebuie să aibă, într-o forma sau alta, un număr. Orice confirmare se referă la un număr de bloc. Numărul de blocuri ce pot fi transmise fără confirmare este limitat fie de protocol (la 7 în HDLC), fie de calitatea liniei, astfel încît să se mențină un echilibru între numărul de retransmisii și timpul de așteptare a unei confirmări. În cazul menținerii unor statistici de erori, pot fi adoptate strategii adaptive.

Dezavantajul principal al ARQ continuu este necesitatea menținerii în memorie a tuturor blocurilor transmise, pîna la primirea confirmării.

Trebuie remarcat că toate protocoalele ce acceptă ARQ continuu ca metodă de retransmisie pot fi utilizate și în ARQ cu așteptare.

Un tip de eroare care nu a fost tratat este cel de time-out pe linie. Această eroare poate avea multiple cauze, de la pierderea accidentală a unor caractere funcționale, pînă la căderi de tensiune sau întreruperi de linie. Pentru tratarea time-out-ului, sînt descrise în capitolul 5 cîteva metode de recuperare (refacere) legatură.

8. PRODUSE PROGRAM

8.1. BREVIAR DE PRODUSE PROGRAM DE TELETRANSMISIE

Prezentăm în continuare produsele-program de teleprelucrare care există în momentul de față catalogate în BNP sau sînt furnizate de către producatorii de echipamente de calcul, pentru diversele familii de calculatoare românești. [36], [37], [38], [40], [41]

8.1.1. Calculatoare FELIX C-256/512/1024

ARIEL

Introducerea interactivă a lucrărilor de la terminale mod caracter și/sau mesaj.

Permite utilizarea următoarelor tipuri de terminale:

- mod caracter (DAF 1001, DAF 2010, CENTRONICS, VDT 40, DAF 2015, DAF 2020, etc.)
- mod mesaj (IRISCOPE 300/1000, TELEROM 3V1, DAF 2010, etc)
- mini-microcalculatoare cu emulator TMM-VU

Oferă utilizatorilor:

- servicii de editare a textelor sursă în regim 'on-line'
- sistem de introducere-extragere interactiva a lucrărilor prin interfață cu SYMBIONT-ul
- sistem de dirijare a rezultatelor execuției spre calculator sau spre stațiile TELESYMBIONT active
- sistem de comunicație inter-utilizatori ARIEL sau între utilizatori și operatorul central sau stațiile TELESYMBIONT
- procedură de contabilizare pe conturi
- sistem de autodocumentare
- interpretor interactiv de BASIC
- procesor de texte
- facilități de conectare logică la un minicalculator a utilizatorilor aflați la terminale gestionate de ARIEL

AMELIE

Metodă de acces la o linie de comunicație în protocol BSC din limbaje evoluate.

Permite programatorilor să realizeze în limbaje de nivel înalt aplicații utilizînd transfer de informație în protocol BSC1 (transfer de fișiere, etc).

Utilizează modulul de acces pentru protocol BSC realizat sub SGT (SRBSC).

AMI

Program pentru gestiunea terminalelor display de tip asincron.

Permite utilizarea terminalelor din programe scrise in limbaje evoluate . Asigură:
- gestiunea rețelei de comunicație, controlul comunicației;
- posibilitatea lucrului cu mai multe terminale;
- posibilitatea testării off-line a programelor;
- comunicarea între operatorul DAF și operatorul de la consola FELIX.

COMINT

Permite consultarea în regim de teleprelucrare a fondului de date privind consumul cu ridicata în vederea raportărilor operative

CONECT

Program de interconectare la distanță a doua sisteme tip FELIX C

Suportul software pentru interconectare folosit este SGT, folosindu-se procedura TMM-UC de tip 2. Programul realizează:
- transfer de fișiere secvențiale pe bandă;
- transfer de lucrări (fișier *1);
- transfer de fișiere de ieșire (*2).

DISCOP

Afiseaza și modifică zone de disc, prin operare interactivă de la un terminal sincron, protocol TMM-VU

MARC-F

Produs program pentru conectarea calculatoarelor FELIX-C la rețeaua națională de calculatoare

Permite comunicația simultană între mai multe programe de pe calculatoare diferite conectate la rețeaua nodala RENOD. Pentru programator se oferă o interfața tip "datagram" și "conexiune logică".

MEMSCOP

Permite afișarea interactivă a unor zone de memorie de la un terminal sincron, protocol TMM-VU

MTSS

Mini-sistem pentru lansarea interactivă a programelor de aplicație de la terminale teletype-compatibile și schimbul de informații între programe și operatorul de la terminal pe durată execuției lor.

MTSS V3 acceptă urmatoarele tipuri de terminale:
- mod caracter (CENTRONICS, DAF 2010, DAF 1001, etc);
- mod mesaj (DAF 2010, TELEROM 3V1);
- microcalculator M18, M18B cu emulator de terminal greu MTSS (LARES T-18 V3);
- CD80 cu emulator LARES T-18 V3.

REJET

Program de test (sub sistemul de operare SIRIS 3) al componentelor unei stații RJE (compatibilă cu terminalul greu MITRA-15) cuplată la un sistem FELIX.

SATEM

Subsistem pentru aplicații ale terminalelor cu ecran, mod mesaj.

Permite utilizarea facilitaților de teleprelucrare în limbaje evoluate (COBOL, FORTRAN, etc.); e compus din module de interfață și module anexe. Admite configurații de 1-8 linii a 1-16 terminale compatibile cu IRISCOPE 300. Permite:
- descrierea și adaptarea configurațiilor de teleprelucrare;
- schimbul de mesaje între calculator și terminale.

SOCRATE

Sistem de gestiune baze de date.

Asigură următoarele funcții:
- definirea și structurarea bazei de date printr-un "limbaj de descriere" propriu sistemului;
- crearea bazei de date prin încărcare "în batch" plecînd de la fișiere clasice sau conversațional (cu date transmise de la consolă);
- exploatarea bazei de date permite actualizarea bazei de date ("în batch" sau conversațional), interogarea punctuală sau masivă a informațiilor.

SPARTE

Sistem de gestiune a terminalelor mod mesaj.

Colecție de rutine, adăugate programului utilizator la linkeditare, care realizează funcții de:
- gestiune a terminalelor;
- gestiune tranzacții;
- paginare a mesajelor (structurarea în 'imagini ecran').

E recomandabil pentru rețele de maximum 64 de terminale și aplicații gen: consultare de fișiere, culegere/introducere de date

STRATEGE

Monitor de teleprelucrare tranzacțional

Asigură următoarele funcții:
- detecția și corecția erorilor de transmisie;
- gestiunea unei rețele de terminale - mod mesaj, mod caracter sau rețea mixtă, cu concentratoare, difuzoare;
- gestiunea tranzacțiilor;
- gestiunea fișierelor utilizator, cu protecție împotriva incidentelor hardware și software;
- analiza transferului în rețea, furnizarea de statistici.

TELECOB

Interfață COBOL-SGT pentru terminale de tip teletype și DAF 1001.

Program ASSIRIS folosind SGT pentru a permite programatorilor în COBOL gestiunea operațiilor pe un terminal tip TTY sau DAF 1001.

TELECTC

Program pentru testarea funcționalității unei rețele de terminale asincrone sub sistemul de operare SIRIS 3.

TELEJOB

Program pentru exploatarea terminalelor RJE în protocol TMM-RB.

TELEMINI

Produs software pentru interconectarea sistemelor FELIX cu minicalculatoare din familia CORAL-INDEPENDENT.

Pachetul cuprinde:
- driverul pentru cuplorul de teletransmisie al minicalculatorului, adaptat pentru protocol TMM;
- protocoalele TMM-UC și TMM-RB, ca module integrabile în programele utilizator pe FELIX și mini;
- o serie de aplicații standard: dialog între operatorii de la consolele calculatoarelor, programe de test al comunicației, transfer de fișiere între calculatoare.

Utilizatorului i se oferă pentru dezvoltarea de aplicații de interconectare, o interfață de programare din COBOL sau limbaje de asamblare.

TELESYMBIONT

Extensie a SYMBIONTULUI pentru preluarea de la distanță a lucrărilor de la terminale grele în protocol TMM-RB.

Permite SYMBIONTULUI să prelucreze lucrările introduse la distanță și să returneze ieșirile la stația emitentă sau alte stații.

Operatorul de la terminal are la dispoziție ordinele SYMBIONT pentru conversia de intrare și de iesire, precum și pentru vizualizarea stării lucrărilor; în timpul execuției job-urilor sale, se afișează de asemenea jaloanele de exploatare.

Poate funcționa în cod EBCDIC sau ASCII.

TSERVICE

Program pentru realizarea unor funcții de 'service' în regim de teletransmisie de la un terminal mod caracter.

TSYBSC

Variantă a TELESYMBIONT-ului care folosește pentru comunicația pe linie protocolul BSC1.

8.1.2. Minicalculatoare din familia CORAL-INDEPENDENT

CAMELEON

Suport hardware și software integrat de prelucrare distribuită, implementat conform recomandărilor ISO-OSI.

Nucleul software integrator, replicat pe fiecare echipament de calcul din sistem, furnizează primitive pentru gestionarea "activităților" (unități de bază de construcție, protecție și distribuție) la nivel sistem și aplicație.

Se admit în configurația distribuită sisteme de tip mini (CORAL, INDEPENDENT) sau micro (M-18B, M-118).

EMMA

Emulator de stație ARIEL.

Implementează protocolul TMM-VU extins pentru stații ARIEL, permițind utilizatorilor să lucreze sub ARIEL de la terminale asincrone conectate la minicalculator și să transfere fișiere pe cartele, imprimantă, bandă magnetică, disc de pe minicalculator din/în zone ARIEL.

De asemenea, oferă suportul necesar conectării logice la minicalculator a utilizatorilor aflați la un terminal legat la FELIX și conectat la ARIEL.

HCnet

Suport software pentru o rețea heterogenă de calculatoare.

Cuprinde următoarele componente:
- PASS: software pentru comutare de pachete
- Hsoft: suport pentru conectarea calculatoarelor de tip 'host' la subrețeaua PASS. Este posibilă conectarea unor calculatoare de tip CORAL/INDEPENDENT, FELIX-C și comunicația între aplicații din oricare tip de calculator
- Fsoft: software pentru transfer de fișiere între host-uri, software pentru serviciul AT - acces terminale asincrone la orice calculator CP din rețea.

MARC-M

Produs-program pentru conectarea calculatoarelor de prelucrare de tip CORAL-INDEPENDENT la rețeaua natională de calculatoare.

Similar cu produsul MARC-F.

MESS/WMO

Sistem de comutare de mesaje pentru meteorologie.

Este un sistem dedicat realizat pe baza specificațiilor din "Manual on the Global telecommunication System" WMO No.386. Comunicația se realizează pe linii sincrone și asincrone, folosind protocoale specifice WMO. Sistemul asigură:
- controlul erorilor de transmisie
- culegerea de date de la terminale tip TTY și telex, precum și de la stații automate
- distribuirea de mesaje în acord cu programe pre-determinate
- schimb de mesaje între centre locale și centre de comutare WMO
- construirea de buletine hidrometeorologice din mesajele recepționate, repetarea mesajelor la cerere

MININET X.25

Pachet de programe pentru nodurile unei rețele cu comutare de pachete care asigură conectarea DTE - DCE utilizînd protocolul X.25, compatibil cu DECNET 11M V2.0.

Facilități și avantaje oferite:
- viteza de transfer la 4800 bps;
- arhitectura de rețea adaptabilă;
- sincronizare pe bit și rata de erori redusă.

Pachetul poate fi implementat atît cu cuploare sincrone de tip DUP 11 cît și cu cuplorul specializat FPX25.

2780M

Emulator de terminal greu IBM2780

Permite transferul fișierelor între minicalcultor și FELIX C (TSYBSC, AMELIE), I100, CORAL (2780M) și microcalculatoare. Lucrează în protocol BSC 1.

NETX25

Sistem de interconectare a nodurilor dintr-o rețea deschisă, pe baza Recomandării OSI.

Asigură interfața X25 DTE, avînd urmatoarele componente:
- driver de rețea orientat pe bit pentru controlul nivelului fizic, utilizînd doua feluri de interfețe: cuplor tip DUP 11 și cuplor microprogramat FPX25;
- proces de rețea pentru controlul legăturii de date - nivelul 2 X.25 utilizînd procedura LAPB;
- nivelul serviciilor de rețea care implementează nivelul 3 X.25 pentru circuite virtuale;
- task auxiliar ACP care implementează nivelul de transport și sesiune;
- pachet de programe pentru trasarea mesajelor de nivel 2 (X25TRC).

Facilități oferite de NETX25:
- comunicație task-task;
- comunicație de mesaje;
- transfer de fișiere și acces la fișiere de la distanța;
- execuție de fișiere de comenzi la distanță.

RMC

Pachet de programe pentru o rețea omogenă de minicalculatoare.

Oferă utilizatorilor:
- suport de comunicație inter-taskuri
- posibilitatea execuției de fișiere de comenzi în oricare din nodurile rețelei
- transfer de fișiere intre noduri
- dialog între terminalele conectate la noduri diferite
- posibilitatea întreținerii și supravegherii rețelei

ROMX-25

Pachet de programe care implementează procedurile și formatele la nivel fizic, legatura de date și pachet pentru minicalculatoare operind în conformitate cu recomandarea X.25 a CCITT; implementarea este realizată în context rețea:
- pentru primul nivel pachetul cuprinde un modul care interacționează direct cu interfața de comunicație
- pentru legatura de date, LAPB asigură transferul secvențial de date, fără erori, pe linia de comunicație care unește cele două stații echilibrate
- nivelul pachet, așa cum a fost realizat, are la bază conceptul de circuit virtual, informația de la nivel superior, structurată în pachete, realizînd multiplexarea unui număr de circuite virtuale pe o singură legatură de date.

SICOMM

Sistem de comunicație mini-micro.

Comunicația se realizează între taskurile din nodurile unei rețele de mini și micro, prin mesaje de lungime variabilă, cu posibilități de sincronizare prin indicatori de evenimente.

Transmisia se face pe legatură serială asincronă (o linie pe micro, 1-8 linii pe minicalculator), cu verificare de paritate pe mesajele transmise.

SITRA

Sistem de gestiune a tranzacțiilor pentru minicalculatoare.

Este un sistem-mediu de programare tranzacțional pentru minicalculatoare din familia CORAL, INDEPENDENT. Răspunde la cele 3 deziderate ale unui sistem tranzacțional:

- gestiune programe prin intermediul unui monitor tranzacțional;
- gestiune fișiere utilizator prin oferirea unor funcții de acces concurent la fișiere (tehnici de lucru tranzacțional pentru asigurarea coerenței și integrității datelor);
- supervizare rețea de terminale și concentratoare:
  - asincrone TTY (display pentru videoformate),
  - asincrone IBM 2660 (DAF2010 sau RIAD),
  - sincrone în protocol BSC3 (IBM 3275, DAF2010-BSC),
  - sincrone IBM 3270 (EMIX80 - emulator pe seria de microcalculatoare M18),
  - alte minisisteme cu această componentă SITRA.

SITRA oferă servicii suplimentare de tipul:

- emulator de unitate de control comunicație IBM 3271 multiplă (conectabilă simultan la 4 calculatoare de prelucrare - IBM, FELIX C, minisisteme sub SITRA);
- emulator de terminal RJE IBM 2780 (2780M), conectabil la IBM, FELIX C (TSYBSC, AMBIIE), mini (2780M) și microcalculatoare cu emulator IBM;
- Conectare într-o rețea omogenă de minicalculatoare sau conectare într-o rețea eterogenă.

SPTC

Concentrator de terminale pentru concentrarea datelor recepționate din rețeaua secundară pe linii asincrone și sincrone, prelucrarea lor la nivelul minicalculatorului și transferul în rețeaua primară către un calculator host emulînd un IBM 3270.

Se asigură în 3 variante:

- sistem local care concentrează datele recepționate de la terminale asincrone simple tip CENTRONICS sau DAF printr-un protocol de casă cît și terminale sincrone conectate prin protocol BSC-3 sau terminale asincrone conectate prin protocol ASYPAC; terminalele simple funcționează în mod. linie sau multilinie;
- sistem transparent care asigură transferul datelor din rețeaua secundară către un calculator gazdă de tip minicalculator, ROBOTRON sau IBM 360/370;
- sistem conversațional care integrează primele două variante.

Asigură interfața utilizator pentru programe MACRO-11 și FORTRAN cît și posibilitatea integrării într-o rețea de minicalculatoare.

TELE

Pachet de programe pentru interconectarea minicalculatoarelor romanești pe baza unei comunicații bit-sincrone. Componenta principală o reprezintă procedura LAPB din protocolul HDLC, implementată ca un task ACP; o altă componentă o reprezintă cea destinată transferului de fișiere între sistemele interconectate.

TELEMINI

Este prezentat în secțiunea pentru calculatoare FELIX.

8.1.3. Microcalculatoare din familia M-18, M-118

CAMELEON

Este prezentat în secțiunea pentru minicalculatoare.

EMIX-80

Emulator de unitate de control al comunicației IBM 3270 pentru microcalculatoarele M18, M18B, M118.

Poate fi instalat, la cerere, pe orice alt microsistem bazat pe INTEL 8080 sau ZILOG 80.

Gestionează terminale asincrone TTY sau display cu mod de lucru paginat (pentru videoformatări) precum și terminale de tip receive-only. Protocolul utilizat pentru legătura cu CP este sincron, polling/selecting BSC3.

Emulator de terminal IRISCOPE 300 (I300)

Implementează protocolul TMM-VU , precum şi funcţiile de terminal (poziţionări pe ecran, coduri de control, etc) specifice pentru IRISCOPE 300

Emulator de staţie ARIEL (TMM-XA)

Implementează protocolul TMM-VU extins pentru staţii ARIEL, permiţind utilizatorilor să lucreze sub ARIEL de la terminale conectate la micro, precum si transferul de fişiere din zonele ARIEL pe imprimanta microcalculatoruiui si de pe cartele citite la micro in zone ARIEL.

HERCULES

Familie de sisteme de operare pentru microcalculatoare cu resurse distribuite.

Asigură conectarea unui microcalculator printr-o linie asincronă la un minicalculator gazdă.

Facilitaţi asigurate de HERCULES:

- acces bidirectional, la fişiere FILES-11 de pe volumele disc din minicalculator;
- transfer de mesaje intre mini si micro;
- acces la resursele reţelei de minicalculatoare;
- generare si exxploatare de date specifice microcalculatoarelor pe volume virtuale din minicalculator.

Variantele familiei HERCULES:

Hr/e - sistem de operare pentru microcalculator independent

Hr/s - sistem de operare pentru orice tip de microcalculator pe 8 biţi, care asigură conectarea ca terminal inteligent la minicalculator;

Hr/d - sistem de operare pentru accesul de la microcalculator intr-o reţea de minicalculatoare.

LARES T-18

Sistem pentru exploatarea microcalculatoarelor FELIX M-18 in regim de teleprelucrare (functii de RJE sub SIRIS 3, protocol TMM-RB) şi pentru realizarea unor funcţii utilitare locale: conversii locale (între perifericele din configuraţie), listare banda magnetică, rezumat banda magnetică; permite ca intrare: banda magnetică, cititorul de cartele si discul flexibil, iar ca ieşire: imprimanta, banda magnetică si discul magnetic; are implementat si protocolul TMM-VU.

LARES **T-18 V3** are emulatoare TMM-RB, TMM-VU si funcţii speciale pentru conectarea sub MTSS V3; NU foloseşte fişiere pe disc flexibil.

REBUS

Program de control al terminalului RCD 7273 de tip RJE.

Permite emularea funcţionării unui terminal greu standard, cuplat la calculatorul FELIX-C şi exploatat sub sistem SIRIS 3 cu SYMBIONT şi TELESYMBIONT.

RJE

Funcţii de RJE pe minicalculator (protocol TMM-RB). Permite ca intrare cititorul de cartele şi ca ieşire imprimanta.

RJE-MT

In plus faţă de RJE permite lucrul cu banda magnetică.

SICOMM

Este prezentat în secţiunea pentru minicalculatoare.

8.2. UTILIZAREA PRODUSELOR PROGRAM IN INTERCONECTAREA ECHIPAMENTELOR DE PRODUCŢIE ROMÂNEASCĂ

Tipurile de aplicaţii pentru care se pot folosi produsele program de teletransmisie existente în fondul B.N.P. sînt următoarele: transfer de fişiere, sisteme tranzacţionale, sisteme conversaţionale, funcţii RJE, sisteme distribuite şi utilitare.

Posibilităţile de interconectare echipamente şi utilizare produse sînt date în tabelele 8.1. si 8.2.

Tabelul 8.1.

		FELIX	MINICALCULATOR	MICROCALCULATOR
APLICATII DE TRANSFER DE FISIERE	FELIX	AMELIE CONNECT TELEMINI		
	MINICALCULATOR	ARIEL-EMMA TELEMINI NCnet AMELIE-2780M	TELEMINI ROMX-25 MESS/WMO NETX25 MININET X25 2780M	
	MICROCALCULATOR	ARIEL-THMXA TELEJOB-LARES ARIEL-LARES V3 TELEJOB-RJE TELEJOB-REBUS TELEJOB-RJE MT TSY-LARES TSY-RJE TSY-REBUS TSY-RJE MT MTSS V3-LARES V3	SICOMM CAMELEON HERCULES	CAMELEON HERCULES

Tabelul 8.2.

	SISTEME TRANZACTIONALE	SISTEME CONVERSATIONALE	FUNCTII RJE	SISTEME DISTRIBUITE	UTILITARE
FELIX	STRATEGE	ARIEL AMI COMINT MTSS SATEM SOCRATE SPARTE TELEJOB	TSY TSYBSC TELEJOB	MARC-F TELEMINI	DISCOP MEMSCOP REJET TELECTC TSERVICE
MINICALCULATOR	SITRA	SPTC		MARC-M CAMELEON TELEMINI TELE ROMX-25 MININET X25 NETX25	2780M
MICROCALCULATOR			I300 LARES RJE RJE-MT REBUS	CAMELEON HERCULES	LARES T-18 V3

ANEXĂ

PROCEDURA DE ÎNCHIRIERE LINII TELEFONICE DE LA D.G.P.Tc.

Pentru utilizarea instalației pentru transmiterea de date pe rețeaua publică de telecomunicație, este necesară o adresă în scris către direcțiile județene de poștă și telecomunicație sau către direcția de telecomunicații a municipiului București, după caz. [M]]

Această cerere trebuie să precizeze următoarele:

- denumirea unității beneficiare
- adresa unității
- amplasarea echipamentelor de transmisie de date (pe camere)
- tipul legăturii (comutată, închiriată)
- caracteristicile tehnice ale instalației
- viteza de comutație
- data la care se dorește darea în funcțiune a instalației
- numărul și tipul circuitelor telefonice

Pentru a se putea conecta la rețeaua de telecomunicații a instalației de transmitere de date este necesar ca beneficiarul să execute următoarele operații:

1. Să ceară la Direcția generală a poștelor și telecomunicațiilor avizarea modemului (care se acordă imediat pentru echipamente cunoscute). Structura acestui aviz este următoarea:

AVIZ 1/nr..........

pentru procurarea aparaturii
pentru transmisia de date ce urmează a se conecta la
rețeaua de telecomunicații publică

D.G.P.Tc. avizează favorabil procurarea de către întreprinderea .... cu adresa în str. .... nr. .... localitatea .... a aparaturii .... tipul .... fabricat de .... la viteza de ....

Prezentul aviz este valabil pentru procurarea aparaturii ce urmează a fi conectată la rețeaua publică de telecomunicație.

Data ....

D.G.P.Tc.
Director general,

2. Să solicite abonament pentru postul de transmitere date care se face lunar sau pe fracțiuni de lună pînă la 15 zile. Serviciul de transmitere date se taxează pentru utilizare de 8 [illegible] și pentru fiecare oră în plus.

3. Să solicite (unde este cazul) închirierea circuitelor telefonice urbane sau interurbane. Taxarea acestora se face pentru fiecare kilometru sau fracțiune.

4. Să solicite direcției județene de poștă și telecomunicație executarea legăturii și măsurătorilor pentru încadrarea circuitelor de date în norme. Confirmarea realizării tehnice a legăturii este prezentată în avizul următor:

AVIZ 2/nr. ....

pentru realizarea legăturii pentru transmisia de date

La cererea întreprinderii .... se avizează realizarea legăturii pentru transmisia de date pentru aparatura instalată în str. .... nr. ..... etaj ...... camera .... Aparatajul va fi folosit la viteza de ...., pe rețeaua comutată sau inchiriată. Tipul modemului este .... fabricat de .... și va funcționa pe doua sau patru fire.

Prin prezentul aviz se asigură timp de 12 luni separarea perechilor la amplasamentul de mai sus, dumneavoastră urmînd să achitați lunar taxele de abonament .... lei și închirieri de circuite ...... lei.

La terminarea lucrărilor, vă rugăm să adresați, în scris, verificarea lucrărilor, în vederea eliberării avizului de punere în funcțiune.

Data .... D.J.P.Tc. - D.T.M.B.
Director

5. La terminarea de către beneficiari a lucrărilor de instalare a aparaturii pentru transmisia de date, conectarea liniilor de telecomunicație la modem se realizează de personalul subordonat Direcției generale a poștelor si telecomunicațiilor care, după o prealabilă verificare a modului în care este realizată lucrarea, eliberează avizul de punere în funcțiune:

AVIZ 3/nr. ....

pentru punerea in funcțiune a instalației pentru transmiterea de date pe rețeaua de telecomunicație publică

În urma verificării modului de execuție a lucrărilor la abonatul . ... din str. .... nr. ...... din ....

Se constată că au fost respectate prevederile avizelor anterioare nr. 1/........ si nr. 2/........ precum și prescripțiile tehnice.

Ca atare se autorizează punerea în funcțiune a instalației.

Avizul poate fi retras în cazul cînd beneficiarul face modificări în instalație, neavizate de D.J.P.Tc. - D.T.M.B.

Data ...........D.J.P.Tc. - D.T.M.B.
Director

BIBLIOGRAFIE SELECTIVA

1. James Martin — System Analysis for Data Transmission Prentice-Hall, Inc. 1972

2. J.E.McNamara — Technical Aspects of Data Communication DEC 1977,1978

3. Cesar Macchi, J.F. Guilbert — Teleinformatique (DUNOD)

4. W.L.Price, D.W.Davies, D.L.A.Barber, G.M.Solomonides — Teleinformatica Rețele de calculatoare și protocoalele lor Editura Tehnică, 1983

5. A.S.Tanenbaun — Network Protocols, Computing Surveys vol. 13, Nr.3 dec 1981

6. R. des Jardins — Overview and Status of the ISO Reference Model of OSI

8. *** — IEEE Transactions on Telecommunication, 1980

9. A. Petrescu, T. Moisa, N. Țăpuș, ș.a. — Microcalculatoarele M18, M18B, M118 vol 1., Ed. Tehnică, București, 1984

10. M. Stein — Les Modems ,ACTIM France, 1977

11. *** — ISO-Open System Interconnection/Basic Reference Model

12. IBM — OS/VS BTAM GA27-6980-3

13. IBM2780 — Data Transmission Terminal GA27-3005-3

14. IBM3270 — Information Display System GA27-2749-9

15. STANSAAB — Alfaskop 3500 Data Terminal System, Reference Manual

16. Catalog INTEL — 8251A Programmable Communication Interface 8253 Programmable Internal Timmer

17. *** — Les procedures en Teleinformatique, DATSA/DME/19037, 1975

18. *** — CDC Matrix Printer. Operator, Installation and Reference Manual,1981,1982

19. *** — CDC 9335 Matrix Printer , Operator, Installation and Reference Manual

20. *** — User Manual Model 761 TELEPRINTER CDC (Centronics)

21. *** — CTQM, Manual de funcționare

22. *** — CTC Manual de prezentare, utilizare și instalare, ICE, București

23. *** — M18 Scheme hardware, ICE, București

24. *** — Multiplexor MX80.20, ICE, București

25. *** — Manual de utilizare M18, M18B, ICE, București

26. *** — Echipamente de tehnică de calcul produse în RSR, Indrumar

27. *** — Manual de modemuri : 3M0, 3M1, 3M5

28. *** — FELAS 2505, Manual de utilizare, ICE, București

29. *** — DAF2010, Manual de operare, ICI, București

30. *** — DAF2010, Manual de utilizare (varianta BSC), ICI,Feper, București

31. *** — DAF2015, Manual de utilizare, FEPER

32. *** — DAF2020, Cartea tehnică, FEPER,1984

33. *** — VDT40C, Manual tehnic, ICE, București

34. *** — CORAL, Manual de prezentare, ICE, București

35. *** — INDEPENDENT 100, Manual de prezentare Hardware, ITC, București

36. *** — ARIEL, Manual de prezentare și utilizare, ICI, 1980

37. *** AMELIE, Manual de utilizare, ICI,1982

38. *** EMMA, Manual de utilizare, ICI,1984

39. *** Indrumar pentru exploatarea tehnicii de calcul în regim de teleprelucrare, ICI,1982

40. *** Catalog de produse BNP

41. *** Tehnici şi metode de lucru utilizate în proiectarea sistemelor informatice în regim de teleprelucrare, ICI,1981

42. *** Racal Milgo, Modem MPS 4800

43. *** Data Modem MD6-12, Installation Manual, ARE, 40009, Rev 0.6, 1977

44. *** CCITT, Vol. 8, Avize V

45. *** CCITT, Avize X

46. *** SGT, Manual de programare

47. *** SGT, Specificații de realizare, CII

48. *** PDP11, Peripherals Handbook, DEC

49. *** DEC, Guide to write an I/O driver

50. *** Caiete de rapoarte tehnice, Nr. 1-16, ICI, BNP

51. *** Instrucţiuni de exploatare privind telefonia urbană

52. *** ELITE 2500 Technical Manual, Datamedia Corporation, Pennsanken, N.J.

53. *** HP 2645 Display station, Reference Manual

54. D.W. Davies, D.L.A. Barber — Reţele de interconectare a calculatoarelor, Editura tehnică, Bucureşti 1976

55. *** CLIO Manual de funcţionare, ICE

56. *** DIAGRAM 2030 Manual de utilizare FEPER 1984

57. *** ISM 150 Manual de operare FEPER 1984

58. *** DDCMP Specification. Version V 4.0. DEC 1978

59. ISO Data communication - X.25 packet level protocol for data terminal equipment, ISO/DIS 8208, June 1984

60. *** ROMX25 Version 1.0, Description manual, ICI

61. *** Operaţii de I/O, INDEPENDENT 100, Manual de programare, ITC, 1980, Cod 00500.01 EP/R

62. *** INDEPENDENT 100, Manual de utilizare şi operare, ITC, 1980, Cod 01800.01 EC/R

63. V. Baltac, ş.a. FELIX C-256, Structura şi programarea calculatorului, Ed. tehnică, Bucureşti, 1974

64. V. Baltac, ş.a. Sisteme interactive şi limbaje conversaţionale, Ed. tehnică, Bucureşti, 1984

65. V. Baltac, ş.a. Calculatoare electronice, grafică interactivă şi prelucrarea imaginilor, Ed. tehnică, Bucureşti, 1985

66. *** PAX-106 Procesor de comunicaţie, Manual de funcţionare, ITC, Timişoara, 1985

67. *** NETX25, Manual de operare, ITC, 1984

68. *** NETX25, Manual de prezentare, ITC, 1984

69. *** NETX25, Manual de utilizare, ITC, 1984

70. *** Hercules, Manual de utilizare, ITC, 1985

71. *** Hercules, Manual de prezentare, ITC, 1985

72. *** STPC, Manual de prezentare, ITC, 1985

„FERIŢI-VĂ DE CAPCANE ÎN ANALIZA DE SISTEM" SCURT GHID DE EVITARE A GREŞELILOR UZUALE

după IIASA

Institutul Internaţional pentru Analize Aplicate de Sistem (IIASA) din Laxenburg (Austria), a organizat, în 1977, o discuţie cu specialişti în tehnicile analizei de sistem din mai multe ţări, despre cele mai frecvente şi uzuale greşeli în aplicarea acestor tehnici. Pe baza concluziilor rezultate s-a întocmit un îndrumător (ghid) pentru uzul analiştilor, intitulat „Capcane ale analizei", cuprinzînd prezentarea sistematizată a unor erori posibile în analiza de sistem, care ar trebui evitate atît de către analişti cît şi de către beneficiari. Ghidul a fost editat de „John Wiley & Sons" (Anglia) în 1980. Un compendiu al acestui îndrumar a fost cuprins într-un raport al Biroului Executiv IIASA. Ţinînd seama de interesul pe care îl pot prezenta numeroasele concluzii rezultate din acest raport pentru specialişti şi cadre de conducere la diferite nivele, îl prezentăm, într-o formă rezumată, în acest volum al AMC, continuînd astfel ciclul de „Analiza şi ingineria sistemelor", iniţiat în AMC-45 şi 46 cu „Ghidul analistului".

Cap. 1. Învăţînd din greşeli

Analiza de sistem a devenit un instrument consacrat de lucru. Multe oficii guvernamentale, firme industriale, organizaţii economice internaţionale au introdus-o deja în activităţile lor de cercetare şi luare de decizii. Alţii doar o experimentează, iar unii încă mai caută să vadă cum s-ar putea ajuta cu ea.

Interesul manifestat de acum în lumea întreagă nu este greu de înţeles. Analizarea a tot felul de sisteme a ajutat la rezolvarea unor importante probleme speciale, economice şi ale mediului, făcînd totodată lumină asupra altora, care de abia îşi aşteaptă rezolvarea. Analiza de sistem a fost îndeosebi folositoare în chestiuni complexe, acolo unde obiectivele au caracter conflictual şi unde planificarea viitoare este însoţită de dificultăţi — fiind deci folosită ca instrument ajutător în stabilirea unei politici.

Cu toate acestea, în ciuda succesului şi a interesului suscitat în întreaga lume, analiza de sistem este departe de a-şi fi manifestat întregul potenţial. Unul din motive este că unii factori de decizie care manifestă interes faţă de ea întîmpină greutăţi în înţelegerea esenţei acestei metode şi a modului cum i-ar putea ajuta. Forma de aplicare a ei diferă de la caz la caz, încît unui factor de decizie i se poate părea dificil să înţeleagă cum poate fi adaptată calea de rezolvare a altei probleme la cea care-l interesează (vezi caseta 1 din pag. 318).

Un alt motiv este că uneori metoda dă greş. Tehnicile analizei de sistem nu se aplică niciodată în probleme uşoare, ci doar în cazul unora grele; iar în cele peste două decenii de dezvoltare şi perfecţionare a metodelor de aplicare — prin împrumuturi de tehnologii din alte discipline, şi adaptări la noi domenii, — analiza de sistem a înregistrat şi o serie de insuccese.

Învăţarea din greşeli este, totuşi, o cale esenţială în analiza de sistem. Analistul lucrează, desigur, cu fapte, presupuneri, modele de sistem şi cu concepte ce pot fi parţial verificate. El trebuie să traducă în termeni comuni limbajele diver-

unor scheme de clasificare logică, ci după unele rațiuni subiective legate de avantaje imediate: ușurința obținerii datelor, fezabilitatea estimărilor sau procedurile operaționale ale biroului de strîngere a datelor. Majoritatea datelor „ad hoc" de contabilitate economică sînt rareori publicate, putînd astfel scăpa din vedere celor care ar trebui să le utilizeze. Problemele de interpretare se complică, deoarece elementele constitutive sînt amestecate de către statisticieni care nu au cules personal datele și fiindcă informațiile au fost manipulate, de cîteva ori, pînă să ajungă la analist.

● După ce analistul a obținut datele necesare — fie direct, fie din surse secundare — transformarea lor în informații folositoare creării unui model sau elaborării unei argumentații analitice devine un pas crucial. O asemenea prelucrare necesită calități cu totul diferite de cele necesare punerii problemei sau colectării datelor. Capcanele pîndesc la tot pasul pe calea efortului de a decide care din numeroasele date posibile sînt cele semnificative și care, transformate, să poată fi folosite la modele sau ecuații; iar acestea, ajustate, să se potrivească cu datele folosite. Testele statistice standard ajută la aprecierea calităților modelului aparent potrivit, dar o simplă evaluare statistică nu poate garanta că au fost alese datele cele mai corecte și adecvate.

Caseta 2

Fiți atenți la mijloacele folosite...

Analiza de sistem este încă în evoluție și pînă acum încă nu posedă criterii general admise de acceptabilitate. Complexitatea sa — accentuată de elemente descriptive, prescriptive, evaluative și de motivație, toate intrepătrunse — face astfel de criterii deosebit de evazive.

În cursul acestei perioade de formare, a existat tendința de a aștepta prea mult de la programarea matematică, teoria așteptării, teoria reglării și de la alte asemenea instrumente care au avut succes în alte discipline și în alte contexte.

Mijloacele de investigație ale analizei sînt neprețuite, dar ele nu pot elimina incertitudinea și nu pot înlocui gîndirea profundă, complexă. Mari capcane îi vor aștepta pe analiștii și factorii de decizie care se vor baza prea mult pe răspunsurile primite doar de la mijloacele de investigație utilizate.

Caseta 3

... și fiți atenți la termenii folosiți

Oamenii de știință folosesc uneori cuvinte și termeni obișnuiți, dîndu-le sensuri tehnice ce pot induce în eroare pe nespecialiști. Termenii (la fel ca și mijloacele de investigație) folosiți în analiza de sistem pot altera comunicarea între analist și factorul de decizie.

Este bine cunoscut faptul că judecata subiectivă joacă un rol „cheie" în succesul unei analize. Dar unii din avocații ei consideră că nu se poate obține o precizie științifică în munca lor prin folosirea excesivă a unor termeni ca „optimizare", „cuantificare" și „formalizare".

Lucrările ce descriu metode analitice, folosindu-se prea mult de termeni și formule tehnice, creează de asemenea confuzii. Teoriile și formulele științifice nu pot asigura o selecție cu adevărat critică din noianul de supoziții alternative, date diverse și metode de analiză.

---

3) *Dezvoltarea mijloacelor de investigație și a metodelor*

O serie de capcane obișnuite provin din alegerea și folosirea mijloacelor și metodelor adecvate din largul sortiment deținut azi de analiza de sistem. Riscurile cresc și mai mult prin accentuata dependență a analizei de sistem de alte discipline (vezi caseta 2). Crearea unor modele de anvergură apelează la matematici,

statistică şi la ştiinţe economice. Problemele în discuţie pot lărgi foarte mult sfera disciplinelor implicate, ca, spre exemplu, biologia şi arhitectura. Largul aspect interdisciplinar al acestei activităţi necesită un efort constant de evitare a înţelegerilor eronate. Majoritatea capcanelor legate de mijloacele şi metodele folosite în analiza de sistem provin mai ales din erorile de comunicare.

● Compromisurile realizate uneori între oamenii de ştiinţă, în vederea stabilirii unui limbaj comun, nu aduc cu sine şi o reală înţelegere mutuală. Uneori rezultă un jargon compozit căruia îi lipseşte profunzimea şi care tinde să mascheze ambiguităţile şi diferenţele, subtile şi dificil de reperat. Cînd unele concepte şi tehnici sînt deplasate din contextul lor disciplinar, acestea pot deveni stereotipe, cu limite ce sînt greu de observat de către cei ce urmăresc o aplicabilitate imediată.

● Valoarea ştiinţifică a unui domeniu se presupune uneori a fi proporţională cu conţinutul său matematic sau statistic. Se aşteaptă mult prea mult din partea unor simple cifre. Cu asemenea presupuneri, cuantificarea şi eleganţa algoritmică pot deveni scopuri în sine, în detrimentul unor elemente ce nu pot fi uşor cuantificate. Aceasta poate împiedica înţelegerea mai adîncă a conţinutului problemei.

4) *Alcătuirea argumentaţiei*

Argumentaţia corelează informaţiile cu concluziile. Ea constituie, în mod obişnuit, o combinaţie complexă de consideraţii metodologice, expuneri de fapte, interpretări, evaluări şi recomandări. Dacă în argumentaţie se includ date ce au fost greşit evaluate iniţial, aceasta poate conduce la capcane în elaborarea concluziilor.

● În alcătuirea argumentaţiei, informaţia nu este sinonimă cu evidenţa. Informaţia devine dovadă doar cînd este inclusă într-un punct specific al argumentaţiei ca un mijloc de a convinge oamenii. O selectare nepotrivită a datelor sau modelelor, plasarea lor într-un moment greşit în argumentaţie sau o prezentare neadecvată în faţa auditoriului pot distruge efectul doveditor al informaţiilor.

● Apreciind nivelul acceptabil de acurateţe al unor date folosite ca probe, se vor aplica standarde diferite unor date de natură diferită. De exemplu, în domeniul ştiinţelor naturii, o eroare de 1 miliard de ani este acceptabilă în aprecierea vîrstei Pămîntului, în schimb valoarea unei constante fizice trebuie ştiută cu mare exactitate.

● Moda actuală de a folosi argumentul matematic în mai toate ocaziile duce la tendinţa de a accepta rezultatele numerice drept fapt real, mai degrabă decît evidenţa. Se uită cu uşurinţă cît de relativă este valoarea unor asemenea rezultate transformate în date, modele sau proceduri estimative. Este poate chiar imposibil de stabilit cu certitudine care din aceste transformări poate avea cele mai mari efecte asupra rezultatelor cifrice.

● Valoarea unor modele de mare anvergură luate ca dovadă este adesea îndoielnică. Slaba documentare poate face aproape imposibilă reproducerea rezultatelor de oricine altcineva, în afara creatorilor înşişi ai modelului. Pe de altă parte, o documentaţie completă ar deveni prea voluminoasă pentru a putea fi studiată chiar de un evaluator expert.

5) *Folosirea concluziilor*

Concluziile unui studiu analitic se pot prezenta ca o prognoză, ca o clarificare a unei dispute, ca o recomandare, ca o evaluare a unor orientări în curs, ca o nouă ideie, sau ca o perspectivă diferită faţă de o veche problemă de orientare. Indiferent de natura prezentării, ea se exprimă în termeni abstracţi ca şomaj, sărăcie, sănătate, învăţămînt sau altele. Acest fapt poate duce la capcane în procesul de comunicare şi de punere în aplicaţie a concluziilor, mai ales cînd un termen abstract a fost folosit cu un sens special.

● Analistului s-ar putea să-i scape din vedere faptul că argumentele raţionale singure nu determină luări de atitudine. Persuasiunea este implicată în orice încercare de a sugera un nou punct de vedere, raţionament sau direcţie de acţiune. Ca să fie eficace, analistul trebuie să fie şi un bun avocat, dar totodată el este şi încrezător în virtuţile metodei ştiinţifice, iar această credinţă este, de regulă, asociată cu o anumită desconsiderare a problemelor şi cerinţelor procesului de comunicare şi convingere.

● Procesul de convingere trebuie văzut ca o componentă a analizei. Probabil că ar fi eficient ca studiul să fie divizat în două etape — determinarea recomandărilor de făcut și apoi recomandarea lor cu convingere — dar cele 2 etape trebuie să se desfășoare în cadrul unui proces unic. Cele mai serioase capcane ale aplicării analizei de sistem îi așteaptă pe cei ce separă planificarea de decizie și folosință, sau gîndirea de realizare.

Cap. 3. În vederea obținerii unui bun model

De obicei, dar nu totdeauna, analiza de sistem implică și crearea unui model. Modelarea este un termen ce stimulează analizarea sistemelor. Aceasta se poate face în multe feluri: empiric, matematic, etc. Adesea, mai multe tipuri de modelare sînt combniate într-un singur studiu. Uneori se folosesc chiar și computerele, dar nu totdeauna.

Indiferent cum s-ar face modelarea, este important de reținut că între formularea problemei și modelare există o strînsă corelație. Modelarea trebuie avută în vedere încă de la început, iar acest capitol descrie unele căi prin care această primă fază a analizei poate duce la greșeli.

Pericole în faza de formulare a problemei

Procesul de analiză începe din momentul în care cineva devine nemulțumit de o stare de lucruri prezentă sau programată. Persoana respectivă s-ar putea să nu prea aibă idee unde se află cheia problemei. Pînă la apariția analistului, este posibil să se fi și exprimat păreri privind o anumită versiune a problemei în cauză. Analistul, însă, mai întîi trebuie să identifice din nou care este problema (vezi caseta nr. 4) și apoi să delimiteze scopul propus. Sarcina nu-i deloc ușoară.

Caseta 4

Confuzia între scop și mijloace

Un factor de decizie exprimă dorința de a găsi un amplasament, în districtul său, pentru construirea unui nou și amplu centru de îngrijire a sănătății. Adevăratul său scop — îmbunătățirea serviciilor de sănătate în urbea sa — nu a fost încă identificat. Acest scop poate fi atins mai eficient prin o serie de alte mijloace: prin înființarea mai multor centre mai mici, dar apropiate, de îngrijire a sănătății; prin dezvoltarea de facilități pentru tratamente ambulatorii în spitalele existente, prin încurajarea medicilor de a înființa mai multe grupe de intervenție, sau prin îmbunătățirea serviciilor medicale la secțiile cu mai mare afluență de bolnavi. Dacă, într-un caz ca acesta, analistul nu face decît să caute un amplasament optim pentru un centru de îngrijire a sănătății, înseamnă că el nu va folosi întregul potențial oferit de o analiză exhaustivă, riscînd ca altcineva să pună în lumină un plan mai bun, care să-i facă inutilă toată strădania sa.

El trebuie să găsească o soluție atît acceptabilă cît și practică în contextul restricțiilor economice, politice, tehnologice, organizatorice etc. Acest proces care cuprinde atît formularea, cît și crearea modelului se poate dovedi greșit, din mai multe motive:

● Dorința de a accepta aprecierea beneficiarului asupra situației, combinată cu dorința de a lansa cît mai repede analiza, îl poate determina pe analist să nu acorde atenția cuvenită formulării problemei. Acest lucru poate duce la rezultate care să nu concorde cu scopurile inițiale, sau ca acestea să nu poate fi duse la îndeplinire, în intervalul de timp și cu resursele disponibile.

● Formularea se poate dovedi improprie pentru folosința avută în vedere. De exemplu, în vederea identificării unor orientări alternative, de obicei trebuie comparate mai multe variante, într-un timp scurt și fără prea mari cheltuieli. În

schimb, pentru implementare — ca, spre exemplu, opțiunea pentru introducerea în fabricație a unor vehicule dintre modele asemănătoare, după ce s-a căzut de acord asupra organizării transportului în comun — analiza trebuie să se facă concret și detaliat. În acest caz, alternativele sînt similare, diferențele dintre ele fiind nu de concepție ci doar de detaliu. Cea mai frecventă capcană în acest caz este de a supune studiului un număr limitat de alternative, analizate concret și detaliat, cînd, de fapt, este necesar să fie luate în considerație o mare varietate de orientări alternative.

● Scopurile nu pot fi stabilite independent de mijloacele cu ajutorul cărora trebuie obținute. De obicei, este imposibil a selecta obiective satisfăcătoare fără a avea vreo ideie despre costuri și dificultățile pe care le implică. De exemplu, lansarea unui om pe Lună n-a fost stabilită înainte ca realizarea ei să devină fezabilă.

● Uneori se mai întîmplă și ca factorul de decizie să prezinte o serie de impedimente arbitrare și nefondate. Astfel, unii beneficiari au refuzat să ia în considerație unele alternative doar pentru că, de exemplu, acestea fuseseră propuse de alte persoane din cadrul aceleiași organizații. În acest caz, din nou, acordarea unei atenții corespunzătoare formulării problemei poate preveni serioase probleme mai tîrziu.

● În formularea unei probleme, reușita trebuie adesea testată, cu ajutorul unui indicator „înlocuitor". Un bun „înolcuitor" poate arăta cît de bine a fost atins un obiectiv, dar capcanele pot apare cînd nu i se recunosc limitele. De exemplu, rata mortalității este folosită spre a tatona starea de sănătate a populației. Prin aceasta se obține un criteriu estimativ, dar rata mortalității nu reflectă mulți alți parametri foarte importanți pentru starea de sănătate a populației. Pericole asemănătoare pîndesc și atunci cînd se încearcă măsurarea veniturilor prin cheltuieli, sau compararea calității nivelului învățămîntului primar din diversele districte după nivelul cheltuielilor făcute pe cap de elev.

Alte dificultăți apar și atunci cînd analistul subestimează complexitatea rezolvării problemelor în lumea practică a afacerilor, în contrast cu cea de laborator sau a unei clase de școală. Analistul trebuie să înțeleagă cum tind să prolifereze problemele în lumea reală. Unul din pericole constă în a neglija luarea în considerație a unor consecințe importante. Un alt pericol este să încerci să faci mai mult decît permit timpul și resursele.

● Analistul, de asemenea, trebuie să se ferească de prejudecăți, mai ales la început. Unele se cunosc mai bine, cum ar fi acordarea de preferințe deliberate unor aspecte ale problemei ce sînt mai ușor de apreciat cantitativ. Altele sînt, însă, mai subtile, cum este prejudecata analistului de a-și însuși, mai mult sau mai puțin conștient, preferințele și țelurile grupului pentru care lucrează. Particularismul și familiarismul sînt cauze majore ale unor greșeli de calcul.

Pericole în faza de elaborare a modelelor

Există multe forme de modele posibile: fizice, limbaje naturale, ecuații matematice, programe de computer, etc. Metoda prognozării include judecata rațională, manipulările fizice, aproximarea numerică, simularea etc. Rolul modelului pentru elaborarea unei linii de orientare este de a indica ce se va întîmpla dacă factorul de decizie adoptă un anumit curs de acțiune, iar în unele cazuri de a indica cel mai bun curs de adoptat.

● Analiștii uneori confundă modelarea cu analiza și chiar cu elaborarea orientării (politicii). Modelele nu constituie decît una din treptele analizei. Procesul include depistarea problemei corecte, desemnarea alternativelor de avut în vedere, interpretarea modelului creat și corelarea respectivei interpretări cu problema factorului de decizie. Percolul rezidă în faptul că uneori se acționează ca și cum modelul ar fi mai important decît problema însăși. Într-un caz și mai rău, analistul se concentrează pe ideea de a adapta modelul la situația dată, fără a se mai gîndi la problema însăși, la decizia ce urmează a fi luată sau la nevoile factorului de decizie. Modelul și rezultatele analizei devin, astfel, irelevante și nefolositoare.

● Analistul s-ar putea să acorde o atenţie preferenţială unui anumit procedeu de modelare (optimizarea, dinamica sistemică sau raportul input-output) şi să adapteze problema potrivind-o modelului favorizat. Spre exemplu, simularea poate fi aleasă ca fiind tehnica cea mai ieftină, mai uşor de realizat şi mai bine înţeleasă. Dar modelul realizat se poate dovedi indezirabil, deoarece nu oferă nici o explicaţie la rezultatele observate sau fiindcă este lent, ceea ce poate duce la creşterea costurilor la un nivel inacceptabil.

● Datele incerte despre care se ştie totuşi ceva sau cel puţin se pot face presupuneri cu oarecare siguranţă tind a primi o mai mare atenţie decît datele incerte despre care se cunoaşte mult prea puţin. Datele incerte calculabile sînt eronat privite ca o provocare. Neglijarea datelor incerte mai dificile poate duce la obţinerea unor rezultate slabe, care să creeze noi probleme la punerea lor în aplicare.

● Unele modele sînt astfel proiectate încît să constituie ghiduri relevante pentru problemă. Dar altele caută să frizeze realismul încercînd să simuleze chiar lumea reală în detaliu, cu intenţia de a găsi răspuns la cît mai multe întrebări legate de situaţia dată. De exemplu, s-au făcut încercări de a modela comportamentul unei metropole, a locuitorilor ei şi a autorităţilor sale, în aşa măsură, încît modelul să poată prognoza ritmul de creştere, mişcarea populaţiei, dezvoltarea industrială şi alte schimbări. Apoi, dacă analistul este criticat că a omis aspecte ale situaţiei, el tinde să facă noi adăugiri modelului, cu noi detalii. Dar, asta nu va putea evita alte critici, deoarece, totuşi, sînt mulţi alţi factori importanţi care rămîn neincluşi. Totodată, modelul devine mult prea complex spre a putea fi mînuit cu uşurinţă. Spre a evita asemenea capcane, problema va trebui astfel formulată, încît să determine cu precizie ce anume se va include în model.

● Este deosebit de periculos a presupune că modelul poate fi validat în mod concludent. În cel mai bun caz, este posibil a obţine o bună înţelegere a punctelor mai tari şi mai slabe ale modelului. Este un fapt că el este util pentru a trage concluziile — dar nu poate fi considerat ca ultim cuvînt în legătură cu ce ne poate rezerva viitorul.

● Modelele sînt uneori concepute mai mult spre a satisface criterii academice decît pentru scopuri politice practice. De regulă, cei ce elaborează modele provin din lumea academică şi tot de acolo primesc şi retribuţiile. Din punct de vedere academic, modelul ideal permite un mare număr de raţionamente, imperceptibile prin observaţie directă. El permite obţinerea, dintr-un număr minim de simulări selectate cu grijă, cu costuri eficiente, a unor concluzii ferme şi are capacitatea de a reproduce rezultate bazate pe vechea comportare a sistemului. Din punctul de vedere al factorului de decizie, însă, modelul ideal este relevant, oportun, demn de încredere, dacă la un cost eficient este capabil să ofere date ce pot fi folosite ca bază de acţiune.

● Mulţi creatori de modele consideră că un model de acţiune poate fi făcut îndeajuns de cuprinzător încît să poată modela chiar procesul în sine de elaborare a liniei de acţiune, satisfăcînd preferinţele şi constrîngerile benefiicarului şi mergînd pînă la a se substitui chiar factorului de decizie. Deşi, într-un număr foarte restrîns de situaţii bine delimitate, se poate obţine şi aşa ceva, factorii de decizie nu trebuie să sconteze că, indicînd doar nişte cifre, vor putea să primească în schimb decizii corecte, în chestiuni complexe.

● Ignorarea sau minimalizarea importanţei costurilor efectuării analizei este una din cele mai serioase erori pe care o pot face analiştii. Beneficiarilor le trebuie informaţii realiste cu privire la costurile analizei, la costurile amînării acţiunii, ca şi la costurile realizării alternativei alese. După cum reiese din capitolul următor, costurile sînt cauza multor dificultăţi în aplicarea analizei de sistem.

Cap. 4. **Costurile alternativelor**

Atît analiştii cît şi beneficiarii tind să subestimeze importanţa cunoaşterii costului alternativelor. Dificultăţi serioase se ivesc din neajunsuri care merg de la ignorarea cu totul a costurilor pînă la omiterea unor alternative cu costuri rezo-

nabile. Înainte de a lansa o analiză, beneficiarii ar trebui să cunoască pericolele, descrise mai jos, legate de costuri.

Ignorarea cu totul a costurilor

Deși majoritatea analiștilor evită să nu ia în seamă costurile, eroarea, totuși, se întîmplă îndeajuns de des spre a o discuta. Scopurile pot fi considerate ca vitale, indiferent de costuri, atunci cînd privesc viața, demnitatea umană sau mediul înconjurător. Dar, e o greșală să formulezi asemenea idealuri înalte, uitînd că țelurile și costurile sînt într-o strînsă dependență. Străduințele de a lua în considerație costurile pot facilita promovarea unor alternative fezabile de realizare a scopurilor.

Contrar presupunerilor obișnuite, cu cît scopul propus este mai important, cu atît mai importantă devine analiza costurilor.

Luarea în seamă doar a unora din costuri

Grija cu care sînt colectate datele legate de costuri, complexitatea formulelor de stabilire a costurilor și precauția cu care sînt prezentate proiectele costurilor totale, toate acestea pot abate atenția de la cel mai critic aspect al analizei costurilor: caracterul ei cuprinzător. Unele estimări, bogat susținute cu date impresionante, dar care, de fapt, nu se referă decît la o redusă parte a costurilor totale, creează o falsă iluzie de a fi cuprinzătoare.

● Analiștii costurilor ar putea pretinde că trebuie să-și rezume eforturile în limitele lor de competență. Asta, însă, poate uneori să cuprindă numai aspectele ușor comesurabile ale costurilor, care apoi s-ar putea să fie prezentate drept evaluare a costurilor totale. Analiștii cu experiență în domeniul costurilor cunosc mai multe tipuri ale acestora: cheltuielile monetare, alte chetluieli ce se pot exprima în unități monetare, cheltuieli ce pot fi exprimate în alt mod decît în unități monetare și cheltuieli care nu pot fi, în vreun mod plauzibil, cuantificate. Toate aceste cheltuieli solicită atenție.

● Costurile pe termen lung sînt de asemenea importante. Atît analiștii cît și beneficiarii ar putea fi înclinați să neglijeze costurile ce privesc un viitor mai îndepărtat. Analiștilor li s-ar putea părea dificil de identificat și evaluat aceste costuri, iar pe beneficiari s-ar putea să nu-i intereseze viitorul îndepărtat din cauza termenelor relativ reduse ale misiunilor lor politice.

Diversele feluri de cheltuieli

Modul în care costurile alternativelor se corelează cu analiza trebuie chibzuit cu grijă. A ține cont de prea mulți sau prea puțini factori în stabilirea costurilor poate crea o falsă imagine asupra întregului proces.

● Includerea unor costuri nesemnificative în evaluări se întîmplă la fel de frecvent ca și ignorarea unor costuri semnificative. Costurile cu adevărat semnificative sînt doar cele care rezultă din decizia specifică supusă analizei.

● Eșecul în evidențierea costurilor semnificative apare, mai ales, atunci cînd costurile sînt aproximate. De exemplu, cînd se pune problema de a hotărî o eventuală extindere a unui program, evaluarea trebuie să includă costuri variabile și nu fixe. Sau, atunci cînd se pune problema de a decide continuarea sau completarea unui program, evaluarea trebuie să includă costuri crescînde și nu descrescînde. Tot astfel, trebuie diferențiate costurile recurente de cele nerecurente, cînd se pune problema unor modificări ale unui program.

● Oportunitatea și interesul sînt uneori confundate. Dacă-ți zugrăvești exteriorul casei tale în purpuriu îi vei scade din valoare — și va scade și din valoarea casei vecinului tău. În mod asemănător, este tentat a considera că analiza costurilor ar trebui limitată doar la un scop îngust. Dar, indiferent dacă scopul este îngust sau nu, analistul trebuie să fie totdeauna conștient de interesul beneficiarului, sau de lipsa interesului acestuia, din motive de popularitate electorală (spre ex.).

O întreprindere sau o comunitate de oameni are adesea tendința de a ignora alte întreprinderi sau comunități — deficiență pe care analistul de costuri trebuie să facă efortul de a o preîntîmpina.

● Uneori sînt confundate și stadiile unei analize. Analistul se poate rezuma doar la a identifica resursele necesare ducerii la îndeplinire a unui plan. Sau, el ar putea încerca să identifice cele mai atractive alternative de folosire a resurselor, sau, mergînd mai departe, să evalueze beneficiile fiecărei alternative. Dacă diferitele componente ale analizei costurilor au fost extinse de-a lungul diverselor stadii de analiză, ele nu trebuie nici omise, dar nici puse de două ori la socoteală.

Despre diferitele volume de cheltuieli

În tot ceea ce privește cheltuielile, nici o altă capcană nu este mai ispititoare decît aceea de a presupune că banii sau valorile monetare pot fi pur și simplu însumate. Valorile bănești diferă de la o situație la alta.

● Costurile și beneficiile ce provin de la entitățile separate ale unui program nu sînt ușor de comparat. Să zicem că o agenție de stat are de ales între două amplasamente pentru construirea unei centrale generatoare de energie, fapt care nu-i pe placul nici unuia din proprietarii existenți pe fiecare din cele două terenuri. Agenția vrea să compare cheltuielile de compensare față de fiecare din cei doi proprietari ai fiecărui teren. Un studiu apreciază că 3 500 US $ pe an ar compensa pe unul din proprietari, în timp ce pentru celălalt ar fi necesari 4 200 US $ pe an. Dacă agenția intenționează să plătească compensații, devine rezonabilă o comparație directă a cheltuielilor implicate de fiecare din cele 2 terenuri. Dar, dacă nu se pune problema unor compensații, cheltuielile urmînd a fi acoperite de către proprietari, comparația nu mai are nici o rațiune. S-ar putea ca nivelul veniturilor unuia din cei doi proprietari de teren să fie considerabil mai mare decît al celuilalt, ceea ce înseamnă că valoarea pierderilor determinate de construirea uzinei pe terenul său, va fi mult mai mică în comparație cu veniturile celui cu venit mare, decît asupra celui cu venit mai mic.

● Mijloacele de determinare a deprecierilor valutare au limite care nu sînt totdeauna evidente. Analizele de costuri efectuate pentru agențiile de stat sînt întocmite cu ajutorul unor jaloane care măsoară gradul de depreciere a valutei: rata dobînzii principale, rata datoriilor guvernamentale, valoarea estimativă a eficienței secundare a capitalului, precum și listele de prețuri cu amănuntul. Fiecare din aceste mijloace are slăbiciuni și avantaje caracteristice și poate afecta în mare măsură analiza costurilor.

● Analiza costurilor poate fi afectată în mod substanțial atunci cînd sînt făcute cheltuieli înainte, sau cu întîrziere față de termen. Aceasta rezultă parțial din deprecierea în timp a valutei, precum și datorită impedimentelor bugetare sau de altă natură, apărute în cadrul agenției finanțatoare. Un exemplu simplificat sugerează cum interacționează factorii implicați. O agenție amînă reparațiile ce trebuiau făcute la acoperișul unei clădiri. În anul următor, cheltuielile de reparare a acoperișului cresc cu 50%, din cauza stricăciunilor ce s-au adăugat datorită scurgerilor ocazionate de ploi, în decursul lunilor de amînare, nemaivorbind de cei 10% cu cît s-a depreciat valuta în decursul anului scurs. Și s-ar putea ca agenția să mai amîne încă o altă cheltuială similară, care să implice o creștere a costurilor chiar și numai cu 25%, în decurs de un an. Lăsînd acum alte considerații la o parte, era bine să repare acoperișul la timp și să amîne alte cheltuieli. Dar, dacă ar fi să ascultăm părerea unui analist atent și cu un înalt nivel de autoritate, s-ar fi dovedit mult mai practic să se efectueze o rocadă de fonduri și să se facă la timp ambele cheltuieli; ceea ce ar fi dus la economii.

Însumarea costurilor

Analiza costurilor trebuie privită ca o analiză a unor alternative. Dacă nu, două feluri de capcane pîndesc pe cei ce le ignoră:

● Costurile care de obicei nu pot fi ușor incluse în calcule monetare sînt adesea neglijate. Pericolul apare cînd atenția este orientată nu spre costuri (cheltuieli) ci spre unitățile monetare folosite pentru măsurarea acestora.

● Analiștii și factorii de decizie caută un singur răspuns la problema costurilor, ceea ce se întîmplă foarte rar. Pericolul rezidă în a nu recunoaște .paleta de alternative care depind, printre altele, de nivelul de competență, de mărimea influenței respectivei autorități și de resursele de care dispune.

Cap. 5. **Ce înseamnă a fi eficient**

Capcanele descrise (de la cap. 2 la cap. 4) privind analiza de sistem sînt de ordin intern. Ele constituie puncte periculoase în care acceptabilitatea analizei poate fi compromisă. Restul acestui studiu se va ocupa de factorii externi — adică de punctele în care o analiză, chiar și bine structurată tehnic, poate pierde din eficacitate.

După cum s-a menționat anterior, prezentarea de față nu se face cronologic. Unele din capcanele expuse mai înainte pot apare în stadiile mai avansate ale unei analize, în timp ce unele capcane externe, discutate în cap. 5 și 6, apar de la început.

Cunoașterea factorului de decizie

O analiză bine structurată tehnic poate da greș datorită unor neînțelegeri ce n-au fost depistate la timp. Daunele își fac apariția progresiv. S-ar putea ca factorul de decizie să considere că activitatea a fost prost îndrumată sau neconcludentă; în acest caz dispare aproape orice speranță ca ea să mai devină eficientă. În aceste cazuri, de obicei, deficiența se dovedește a fi una și aceeași: analistul nu a știut care este aspectul ce prezintă cel mai mare interes pentru beneficiar. Activitatea se poate, apoi, derula prost în mai multe feluri:

● Factorul de decizie s-ar putea să lucreze într-un colectiv în care părerile contează mai mult decît faptele. Realitatea practică a intereselor sale s-ar putea să-i obstrucționeze capacitatea de a distinge între fapte și valori, între interesul personal și cel public, între politică și procesul politic în evoluție, precum și între luarea unei hotărîri și judecata morală. Alternativele propuse de analist trebuie să se potrivească cu opțiunile beneficiarului.

● Analistul s-ar putea să prezinte niște rezultate mai complexe decît este nevoie. Ca să poată fi convingătoare, analiza trebuie să fie clară. Necunoașterea exactă a nevoilor beneficiarului poate duce la un text de prezentare confuz, la argumente vagi și la rapoarte ce oferă mai multe detalii decît trebuie. Aceasta poate face ca o analiză — de fapt bine făcută — să fie respinsă.

● Unele orientări sînt stabilite doar de un singur factor de decizie, dar, cel mai adesea, sînt implicate mai multe persoane. O analiză care ia în considerare doar un singur factor rațional de decizie rar se întîmplă să se dovedească adecvată. Unii analiști explică rezultatele unei politici prin interacțiuni între instituții sociale, diverse grupări și cetățeni care participă, fiecare după putere, la influențarea unor decizii. Alți analiști se concentrează asupra grupărilor organizatorice. Oricum ar fi, în vederea evitării capcanelor ce pot apare la punerea în practică a unei analize, mai totdeauna trebuie să se țină seama de dinamica variatelor tipuri de grupări comportamentale. Analistul și beneficiarul pot colabora pe diverse linii strategice cu scopul de a face munca eficientă, ca, de exemplu, urmărind mai de grabă soluții satisfăcătoare decît optimizări maximale, sau concentrîndu-se pe înlăturarea unor blocaje și adoptarea unor măsuri treptate (vezi caseta nr. 5).

● Nu-i mai puțin adevărat că și atunci cînd se manifestă prea multă înțelegere față de factorul de decizie și de nevoile sale, de asemenea, este posibilă apariția unor pericole. Analistul care ajunge să gîndească la fel ca beneficiarul analizei s-ar putea să adopte necritic punctul de vedere al beneficiarului. Cînd analistul, în mod inconștient, adoptă atît prejudecățile cît și atitudinile necugetate (oarbe) ale beneficiarului, acestea îi vor submina spiritul de detașare științifică,

Caseta 5

Apropiere, da — dar nici prea, prea!

Relația ideală dintre analist și factorul de decizie îi permite analistului să-și mențină obiectivitatea științifică, căutînd, totodată, să privească problema din punctul de vedere al factorului de decizie. Dacă analistul este apropiat de beneficiar, dar nici în prea mare măsură, atunci el își poate pune întrebările următoare, de pe o poziție detașată și avantajată:

- *Care-i scopul urmărit de activitatea mea?*
- *Pe cine încerc să influențez și pe ce cale?*
- *Ce tipuri de proceduri analitice sînt necesare?*
- *Ce căi de abordare a acestor sarcini analitice pot da cele mai bune rezultate?*

Răspunsurile țin mai mult de judecată decît de tehnică — adică de o investigație rațională sprijinită de intuiție, care provine din experiența acumulată în aplicarea analizei de sistem.

Dacă analistul reușește să înțeleagă factorul de decizie și nevoile acestuia, el va ști, încă înainte de lansarea analizei, dacă urmează a fi folosită, mai ales, spre a rezolva o problemă, sau doar pentru a informa asupra ei, atît factorul de decizie cît și pe alții. Factorii de decizie uneori se folosesc de analiză doar spre a se instrui, fără să-i facă analistului cunoscut acest lucru. Știind de la început scopul pentru care se face analiza, analistul poate pune la dispoziție rezultate eficace.

Factorul de decizie trebuie să fie adînc implicat în analiză spre a se asigura succesul acesteia. Cînd analistul și beneficiarul conlucrează strîns, ținînd totuși o distanță respectabilă, beneficiarul poate contribui la îmbunătățirea eficienței analizei, prin urmărirea fiecărei etape a acestuia. În acest caz, el nu va ajunge niciodată, în finalul analizei, să aprecieze rezultatele acesteia după părerile altora sau prin idei preconcepute față de munca depusă.

necesar unui analist. Acesta trebuie să înțeleagă bine motivele ce l-au determinat pe beneficiar să ceară o analiză de sistem, dar nu e obligatoriu să și le însușească.

Capcanele expuse mai sus sugerează cum o analiză de înaltă ținută, din punct de vedere tehnic, poate deveni ineficientă, în sensul punctelor menționate în cap. 2 „Analiza de la A la Z". Ca o ilustrare a căilor prin care o analiză poate fi abătută de la scopurile ei, vom prezenta mai jos cîteva exemple de cazuri reale. Ele au fost aranjate după principalele faze ale analizei, ca în cap. 2.

Depistarea problemei și colectarea datelor

Secretarul Departamentului de Interne al S.U.A. trebuia să decidă dacă să continue construirea Complexului Bonneville din centrul statului Utah. Proiectul privea un larg sistem multifuncțional de irigație și folosințe de apă. Secretarul trebuia să știe dacă să aprobe contractele de construire a barajului de pe fluviul Currant Creek, acesta constituind următoarea etapă în realizarea complexului Bonneville. Toată problema era de stabilire a unor priorități.

Analiștii săi au studiat aspectele și alternativele și i-au prezentat rezultatele, care duceau la concluzia că realizarea completă a Complexului va crea probleme. Alternative mai ieftine și mai puțin dăunătoare mediului înconjurător puteau rezolva cele mai importante nevoi de apă ale zonei respective. Totodată, amplasamentele alternative ar duce la evitarea unor probleme ridicate de drepturile legale ale tribului indian Ute asupra apelor teritoriale, subiect tot mai mult disputat pe plan local.

Atunci Secretarul a dorit să cunoască consecinţele construirii doar a barajului, fără a mai realiza întregul Complex Bonneville. Ar fi oare el util dacă nu s-ar mai executa tunelele, apeductele sau uzinele de pompare? Iar dacă da, şi dacă construirea lui n-ar implica nişte probleme prea serioase legate de cost, de mediul înconjurător şi de dreptul indienilor la ape teritoriale, ar fi oare mai bine să se meargă înainte cu construcţia lui, în timp ce se supun studiului aspectele mai mari ale celorlalte probleme?

Pasionaţi de sarcina evaluării întregului complex Bonneville, analiştii au neglijat să ia în consideraţie şi posibilitatea, ceva mai restrînsă, care-l atrăgea pe Secretar. Sub presiunea unor factori politici de a continua construcţia şi trebuind să facă faţă altor probleme mai importante ale Departamentului său, Secretarul dorise, de fapt, să ştie dacă poate amîna luarea în consideraţie a problemelor mai mari. În cele din urmă a primit răspuns afirmativ. Analiştii nu greşiseră în evaluarea per total a Complexului Bonneville. Greşeala lor n-a fost decît că au scăpat din vedere luarea în studiu a unor alternative de mai mică amploare, dar mai practice.

Dezvoltarea mijloacelor de investigaţie şi a metodelor

Un studiu privind piaţa de vînzare a caselor a fost promovat ca parte a Programului de reconstrucţie a comunităţii oraşului San Francisco. Obiectivul general al analizei era să promoveze o largă gamă de acţiuni publice şi particulare care să ducă la îmbunătăţirea condiţiilor de viaţă în mediul citadin, concentrîndu-se în special pe piaţa de vînzare a caselor. Sarcina analiştilor era să dezvolte un cadru de replică adecvată a operaţiunii printr-o piaţă particulară de vînzare a caselor.

Prima problemă era delimitarea geografică a studiului. Şi-au îndreptat atenţia asupra oraşului San Francisco, ştiinduse că piaţa importantă de vînzare a caselor depăşea teritoriul oraşului. Planurile de dezvoltare a oraşului ar fi fost afectate dacă nu ar fi prevăzut şi aceste probleme. Propunerea de a elabora un plan pentru întreaga regiune era nerealistă, mai ales că obţinerea de către beneficiar a acordului factorilor de decizie din întreaga zonă asupra acţiunii ar fi fost cu mult peste puterile sale.

Următoarea problemă plana şi mai ameninţător asupra studiului: dezacordul cu privire la volumul necesar de detalii. Planificatorii oraşului voiau o analiză detaliată bazată pe un vast volum de date. În schimb, personalul de cercetare operativ dorea o analiză simplă, avînd deja experienţa unor modele simulatoare, scăpate de sub control din cauza aglomeraţiei de date. Analiştii au obţinut un compromis, dar devenea prea simplu de promovat ceea ce cereau planificatorii şi prea complicat ceea ce doreau cercetătorii operativi. Capcana în acest caz se ascundea în modul nepotrivit în care analiştii au prezentat beneficiarului propunerile lor de modele.

Alcătuirea argumentaţiei

Agenţia americană de protecţie a mediului înconjurător dorea să cunoască măsura în care poluarea cu acid sulfuric produsă de convertorii catalitici de pe automobile constituie un pericol pentru sănătatea cetăţenilor, precum şi dacă ar fi motivată o amînare în aplicarea standardelor de emisiuni admise legal.

Expunerea umană la concentraţii de acid sulfuric din aer nu poate fi măsurată direct. Un grup de analişti au folosit modelul de dispersie a monoxidului de carbon spre a aproxima gradul de expunere a pietonilor în vecinătatea şoselelor mai importante prin sondaje la orele cu vîrf de trafic, atît pe vreme bună cît şi pe vreme nefavorabilă.

Calculînd datele obţinute pe model, în urma echipării cu converzori catalitici a unui număr de automobile din patru modele au ajuns la concluzia că riscurile erau mai mari decît foloasele. Beneficiarul a luat apoi o decizie controversată de a slăbi presiunile exercitate pînă atunci asupra uzinelor de automobile privind instalarea convertoarelor.

Analizele ulterioare au scos la iveală că respectivele concluzii s-au bazat pe combinarea unei serii de presupuneri false care au dus la un scenariu improbabil. O cu ottul altă imagine s-ar fi obținut dacă s-ar fi prezentat explicit incertitudinile datelor culese, ale presupunerilor și calculelor făcute pe model și dacă s-ar fi promovat studii de tatonări corespunzătoare. Evaluările realizate mai recent asupra gradului de expunere la acid sulfuric au indicat că prima analiză fusese extrem de exagerată încă din faza de pregătire a argumentației analitice.

Folosirea concluziilor

Un studiu efectuat de Institutul Național de boli mintale din S.U.A. arăta că respitalizarea bolnavilor mintali după externarea anterioară ar putea fi redusă, dacă un lucrător de asistență socială din regiunea de reședință a pacientului i-ar vizita la spital, stabilind cu acesta relații de muncă și participînd la reîncadrarea lor socială.

Analiștii au propus împuternicitului de la Min. Sănătății Publice să pună în aplicare această practică pe întreg teritoriul țării.

În acest caz capcana a provenit din greșeala de a nu lua în considerație problemele ce ar putea apare în cazul punerii în practică a respectivei propuneri, fapt care a devenit foarte clar analiștilor cînd împuternicitul Ministerului le-a trimis răspunsul său la propunerea lor. Ar urma oare ca lucrătorul de asistență socială să aibă aceeași calificare ca și cei din respectivul spital? Ar fi posibil să se mențină aceeași normă ca și a celor din spital de numai 6 pacienți per lucrător social?

Totodată el reținuse că respectivul funcționar la care se referea studiul își avea biroul în apropierea spitalului. Oare toți lucrătorii de asistență din regiuni vor fi dispuși să locuiască, departe de familiile lor, pe perioada vizitării spitalelor? Cine ar urma să suporte cheltuielile de deplasare? Cine urmează să asigure instructajul lor? Oare cum vor fi aceștia priviți de către personalul spitalicesc al serviciului social intern? Iar dacă personalul din spital ar urma să aibă, astfel, sarcini mai puține, nu s-ar putea ca unii să-și piardă locul de muncă?

Împuternicitul a mai reținut faptul că procentul celor care sînt respitalizați era destul de scăzut, și întreba dacă îmbunătățirea propusă va avea într-adevăr efecte de luat în considerare. În încheiere, a adresat o ultimă întrebare pe care, de fapt, nici un analist și nici un factor de decizie n-ar trebui să uite să și-o pună: „Dacă măsura se adoptă, oare va fi cineva care să se poată felicita că a sprijinit și promovat această idee?"

Cap. 6. **Analiza pusă în aplicare**

Analiza de sistem poate fi folositoare chiar și numai prin adnotarea rezultatelor ei, sau acestea pot sta la baza elaborării unei orientări sau a unui program de activitate. Cînd se trece la executarea unui astfel de program, aceasta presupune reașezarea într-o nouă configurație a relațiilor sociale, politice și economice existente. Dacă noua configurație corespunde cu cea avută în vedere și dacă ea a rezultat întocmai din concluziile trase de analiză, se poate considera că procesul de punere în aplicare a reușit.

O orientare ce dă greș nu trebuie neapărat să provină dintr-o greșită aplicare. Oricît de corect aplicată, o acțiune improprie nu va putea da rezultatele dorite. Într-un asemenea caz, punerii în practică i se poate reproșa cel mult că n-a reușit să-i ajute pe cei în drept să sesizeze, măcar cu un ceas mai devreme, greșelile lor conceptuale, spre a mai putea lua măsuri de corectare a lor.

Procesul de elaborare a orientărilor, obiectivelor și acțiunilor trebuie să includă modalități permanente de verificare și evitare a greșelilor. Punerea în aplicare reușită nu trebuie să evite doar capcanele, ci să caute mereu cărări mai bune și poate mai imprevizibile de rezolvare a unor noi cerințe, care poate nici n-ar fi avut cum să fie prevăzute la momentul stabilirii politicii de urmat. Capcanele din faza de aplicare în practică sînt, în general, invizibile la începutul procesului de aplicare.

Procesul de punere în practică implică mulți actori, din care unii caută a se ajuta reciproc, alții, dimpotrivă, urmărind avantaje strategice precum și anumite rezultate finale. Din acest punct de vedere, el seamănă foarte bine cu o serie de jocuri în care rezultatul unora afectează condițiile de joc ale altora. În acest ultim capitol sînt descrise patru tipuri din cele mai frecvente variante posibile de jocuri de implementare, fiecare din ele sugerînd capcane ivite în cursul efectuării analizei:

Resurse deviate

● *„Bani cîștigați ușor"* — Contractele încheiate cu guvernul constituie puncte de reazem pentru multe întreprinderi particulare, în economiile cu piețe libere. Problemele apar atunci cînd guvernul acordă unor parteneri de contract nepricepuți, necalificați, neverificați poziții de răspundere în realizarea unor contracte. Greșeala de a încheia un contract în asemenea condiții este greu de îndreptat. Directorul unui program care n-are puterea și curajul de a-și recunoaște greșeala făcută, va fi permanent șantajat de răuvoitorul partener de contract.

● *„Bugetul"* — șefilor de servicii guvernamentale le plac bugetele mari, iar, acolo unde se practică bugete supraîncărcate, cu cît cheltuiesc mai mult, cu atît speră să mai primească. Cheltuielile ridicate sînt de obicei încurajate de autoritățile înalte. Cheltuielile neproductive sînt cel mai intens stimulate cînd serviciul respectiv are și rol de intermediar financiar, în general în calitate de distribuitor al fondurilor de „ajutorare" pentru alte autorități de stat de la alte nivele. În cel mai rău caz, Agenția intermediară este chiar apreciată, aproape exclusiv, după priceperea cu care reușește să plaseze banii, ceea ce va avea grijă s-o facă cît mai bine.

● *„Acordarea fondurilor"*. Un serviciu finanțat de la buget sau o organizație care nu lucrează pe principii economice va căuta să maximalizeze o donație prin stimularea solicitanților de fonduri și să minimalizeze eventualele constrîngeri prin zădărnicirea măsurilor de supraveghere practicate de Agenția donatoare. Supravegherea este ineficientă mai ales atunci cînd scopurile programului sînt greu de apreciat și evaluat.

● *„Viața ușoară"* — Unii birocrați inventează căi prin care să nu se ostenească prea tare, dar să lase impresia că ar munci din greu. Și cînd muncesc ceva mai mult, în special la aplicarea în practică a unei anumite politici (orientări), o fac într-un mod convenabil și rutinier. Dacă noua orientare impune o ruptură cu practica (rutina), un astfel de individ o poate ușor zădărnici.

● *„Pușculița"* — Presiunile politice exercitate cu scopul de a dona bani cu titlu de patronaj în stînga și-n dreapta nu sînt neapărat un lucru rău, în afară de faptul că resursele financiare impun să fie concentrate spre a atinge un anumit prag de eficiență. Trebuie orientată cu grijă acțiunea de implementare pentru a asigura suficienta concentrare a resurselor în vederea obținerii rezultatelor scontate.

Scopuri deviate

● *„Menținerea liniștii"* — Noile legi privind protecția mediului înconjurător, a sănătății și securității lucrărilor sau alte reglementări sociale creează posibilități zeloșilor să preia controlul asupra aparatului de comandă. Alții, care reprezintă poziții conservatoare, pot interveni spre a neutraliza noile programe sau măsuri de înnoire, instalînd în poziții cheie simpatizanți de-ai lor, stabilind standarde și nivele coborîte ca obiective de atins, slăbind sancțiunile pentru încălcarea dispozițiilor sau introducînd criterii foarte riguroase și dificile pentru demonstrarea și dezvăluirea violărilor. Într-un asemenea climat implementarea este ușor de neutralizat.

● *„Care, pe care"* — Aceasta se întîmplă cînd suporteri ai unor orientări (politici) adverse, fiecare avînd altă părere despre obiectivele programatice ale Agenției, pun mîna, pe rînd, pe conducerea ei. Rezultatul este identic, ca și în cazul anterior: neutralizarea punerii în aplicare a respectivei orientări (politici, linii de conduită).

● *„Sufocarea"* — Dacă un nou program se bucură de succes, sprijinul său politic devine tot mai larg. Apoi, însă, el devine ținta unor interese care nu mai au aproape nimic comun cu obiectivele sale inițiale. Pe măsura ce acestea se îngrămădesc, obiectivele originale ale programului sînt subminate, sau grupul celor ce au sprijinit inițial programul poate fi strivit sub greutatea noilor interese.

Energii risipite

● *„Tenacitate"* — Dacă cei ce participă în procesul de implementare au preferințe diferite asupra rapidității de executare a programului, cei mai grăbiți sînt de regulă vulnerabili la manevrele celor ce doresc întîrzierea sau sînt mai puțin interesați în implementarea rapidă. Marele pericol este ca fracțiunea cea mai dîrză să împingă lucrurile prea departe, astfel încît să-i descurajeze pe toți ceilalți; aplicarea programului se destramă astfel, din lipsă de suport politic sau financiar.

● *„Teritoriu"* — Pînă la un punct, concurența birocratică pentru controlul asupra aplicării unui program poate fi constructivă, sau poate deveni o piedică în calea eforturilor de coordonare a responsabilităților diferitelor agenții colaboratoare la nivel operativ.

● *„Nu-i problema noastră"*. — Acest caz este contrar celui anterior. Agențiile caută să fugă de preluarea răspunderii noului program, care-i privit ca o povară deloc încîntătoare. La fel se întîmplă și atunci cînd o agenție primește noi însărcinări fără o creștere corespunzătoare a resurselor bugetare. Execuția, în aceste cazuri, poate fi amînată ani în șir.

● *„Stingherul"* — Nesiguranța acțiunilor colective a determinat multe Agenții să obțină clauze contractuale care să le permită să se retragă din colaborare, la nevoie. Strategia constă uneori din manevre prin care agențiile să fie determinate să renunțe la opțiunea de retragere — de obicei prin impunerea plății unor sume mari, care să le alimenteze interesul față de proiect și să le descurajeze de a-l abandona. Cînd victima vizată, însă, reușește să opună rezistență acestei strategii, rezultatul tinde să ducă la blocarea proiectului.

● *„Reputație"* — Politicienii caută să-și cîștige reputația de a fi sensibili față de alegători. Analiștii politologi caută să-și cîștige reputația de a fi subtili, cu înaltă calificare și de mare utilitate. Politicienii caută să-și cîștige reputația de a fi perspicaci, discreți, loiali, capabili de a obține un consens și — să sperăm de acum — în stare să evite capcanele obișnuite ale analizei de sistem. Efectele căutării unei reputații nu sînt totdeauna rele. Pericolele apar atunci cînd o pernoasă reușește să convingă pe alții că el face mai bine și mai mult decît face în realitate, precum și atunci cînd, prin modul în care „pozează", demoralizează pe alții care depun, de fapt, o muncă mai constructivă dar mai puțin vizibilă în implementarea unui program.

BAZE DE DATE. PROIECTUL — PILOT RA

Mat. Margareta Drăghici
ITCI

Destinat cunoscătorilor teoriei relaționale a datelor, articolul este un apel la colaborare.

În jurul anilor '70 lumea informatică intuia că se află în pragul unui deceniu „al bazelor de date". Era un moment de entuziasm — izvorît din înțelegerea semnificației profunde a noțiunii — și de speranță în caracterul operațional al conceptului.

Sfîrșitul deceniului este dominat de insatisfacția utilizatorilor. După o perioadă de încercări practice, dirijate prin excelență de intuiție și bun-simț, o contabilitate simplă a arătat disproporția între efortul uman și rezultatele obținute. Au apărut scepticii și denigratorii; literatura de specialitate a oferit sute de pagini de caricatură și necrolog... A apărut însă și atitudinea științifică: realizatorii de sisteme de baze de date au înțeles importanța fundamentului teoretic.

Actul de naștere a unei construcții teoretice riguroase — *teoria relațională a datelor** — fusese semnat în anul 1970, odată cu apariția lucrării lui E. F. Codd „A Relational Model of Data for Large Data Banks" (Comm. ACM, vol. 13, no. 6, June 1970). În cei 15 ani care au trecut, teoria relațională s-a dezvoltat ajungînd să ocupe un loc de prim ordin în știința calculatoarelor· Există, în continuare, probleme deschise, formulate de practică și nerezolvate în cadrul etoriei relaționale... Dar proiectul celei de a cincea generații de calculatoare soluționează problema mecanismelor pe baza elementelor oferite de teoria relațională și de inteligența artificială! Este cea mai bună *recunoaștere a valorii teoriei relaționale.*

Realizatorii de baze de date din România au traversat și ei perioadele de-alungul cărora insatisfacția a luat locul entuziasmului inițial. Cu timpul, grupuri de cercetători din țară au ajuns la recunoașterea rolului teoriei relaționale, la cunoașterea ei și chiar la elaborarea unor soluții originale. Cea ce *apreciem că lipsește în acest moment* este un cadru propice stabilirii „comunicării" între grupurile disparate, atît în vederea confruntării profesionale, cît și în scopul conjugării efortului pentru atingerea unor finalități cu rezonanțe practice.

În anul 1984, în Institutul central pentru conducere și informatică** a fost demarat *proiectul-pilot RA* (*R*elational *A*pproach). Apreciind să

* Autoarea acestui articol apreciază că teoria relațională este singura „teorie" a datelor existentă la ora actuală fără a nega importanța unor lucrări care nu se înscriu în teoria relațională, considerăm că ele pot fi etichetate doar ca „abordări".

** Actualmente Institutul de cercetări și inginerie tehmologică pentru tehnică de calcul și informatică (ICTI).

proiectul este de natură să încurajeze o largă colaborare, prezentăm în cele ce urmează elementele lui definitorii.

1. *Terminologia propusă în cadrul proiectului*

Pentru a înlătura ambiguitățile induse de folosirea abuzivă (în special din motive comerciale) a unor termeni, în cadrul proiectului a fost adoptată o convenție terminologică. Introducerea unor termeni suplimentari destinați să nuanțeze diferențele existente între concepte desemnate în literatura comercială printr-un singur nume, a fost justificată prin evidențierea esenței teoriei relaționale, rezumată mai jos.

Esența teoriei relaționale

a) Spațiul tuturor informațiilor vehiculate în sistem este privit ca o mulțime de relații. Pe mulțimea tuturor relațiilor este definită o mulțime de operații; *mulțimea relațiilor este închisă* față de aceste *operații* (structura matematică astfel definită se numește „Algebra Relațiilor"). Operațiile au rolul de a deriva, dintr-o mulțime dată de relații, noi relații. Este definită noțiunea de completitudine relațională care permite compararea expresivității unui limbaj (destinat formulării cererilor de regăsire) cu expresivitatea Calculului Relațional (limbaj bazat pe calculul cu predicate de ordinul întîi).

b) Este definită noțiunea „structură conceptuală optimă" (teoria normalizării); optimizarea are în vedere minimizarea riscurilor de eroare la actualizarea bazei de date. O serie de anomalii grave care pot să apară la actualizarea colecțiilor nenormalizate de date, generînd stări de inconsistență, este evitată în cazul colecțiilor de relații normalizate.

c) Sînt definite noțiunile „descompunere care prezervă dependențele" și „descompunere fără pierderi față de joncțiune"; fiecare din aceste noțiuni asigură o anumită echivalență de ordin semantic între două colecții de relații.

d) Operațiile cu relații nu fac apel la pointeri, indecși, fișiere inverse, etc. Insistăm asupra acestui element central al teoriei relaționale, neglijat în implementările concrete din motive explicabile pentru stadiul actual al tehnicii, dar uitat în mod nejustificat în cercetările teoretice.

Definiția noțiunii SGBD-relațional

Ee numește SGBD-relațional un sistem de gestiune a bazelor de date cu următoarele proprietăți:

1) Este construit într-o arhitectură în care „nivelul intern" și „nivelul fizic" (în accepțiunea definită de ANSI/SPARC) se confundă (cerința decurge din d).

2) Unitatea de informație cu care se lucrează este „relația" (în conformitate cu a).

3) Colecția de relații care intră în componența schemei conceptuale este normalizată și „echivalentă" semantic cu relația universală definitorie pentru Universul Discursului modelat (în conformitate cu b) și c).

4) Primitivele de acces la date sînt operatorii Algebrei Relațiilor (în conformitate cu a).

Proprietatea 1) nu poate fi asigurată în implementări care fac uz de tehnica de calcul tradițională (în absența unor „mașini de baze de date")

deoarece s-ar ajunge la o degradare a timpilor de acces peste limitele admisibile.

Condiţia 3) nu poate fi asigurată, la ora actuală, prin intermediul SGBD-ului, deoarece în cea ce priveşte algoritmizarea procesului structurării conceptuale nu au fost găsite însă soluţii convenabile (în pofida considerabilelor forţe de cercetare antrenate în rezolvarea problemei). În aceste condiţii calitatea schemei conceptuale scapă de sub controlul SGBD-ului şi nu maipoate fi formulată decît o condiţie mult mai slabă decît 3);

3′) SGBD-ul dispune de un limbaj în care un factor uman — administratorul bazei de date — consemnează rezultatul procesului structurării conceptuale; responsabilitatea schemei conceptuale (sub aspectele b) şi c) revine integral administratorului.

SGBD-uri pseudo-relaţionale şi SGBD-uri cu interfeţe relaţionale

Pînă în prezent sînt comercializate sub eticheta „SGBD relaţional", după tipuri de produse, pe care le vom numi SGBD-uri pseudo-relaţionale şi SGBD-uri cu interfeţe relaţionale.

SGBD-urile pseudo-relaţionale:

— au proprietăţi 2) şi 4);

— au proprietatea 3′);

— nu au proprietatea 1).

SGBD-urile cu interfeţe relaţionale:

— au proprietatea 2) şi 4) numai în raport cu funcţia de interogare;

— în general, nu au proprietatea 3′);

— nu au proprietatea 1).

În concluzie, produsele comercializate nu beneficiază de întreaga putere a teoriei relaţionale. Este important ca acest lucru să nu fie escamotat ci, din contră, pus în evidenţă, pentru ca cercetarea să caute soluţii corespunzătoare.

2. *Definirea direcţiilor de cercetare pentru proiectul RA*

Investigarea tendinţelor pe plan mondial, dar şi a cerinţelor rezultate din practica utilizării bazelor de date, au condus la formularea următoarelor direcţii de cercetare:

1) Elaborarea, verificarea şi compararea modelelor teoretice relaţionale şi a soluţiilor de implementare.

2) Consolidarea unor soluţii şi aplicarea lor în realizarea unor interfeţe relaţionale pentru SGBD-uri existente.

3) Construirea unui SGBD pseudo-relaţional.

4) Definirea unui SGBD relaţional experimental; implementarea lui; evaluarea performanţelor în vederea evidenţierii „punctelor critice".

5) Definirea cerinţelor pentru o maşină de baze de date (pe baza concluziilor cercetărilor enunţate la punctul precedent).

6) Elaborarea de componente ale unei biblioteci de programe pentru proiectarea automată a structurii conceptuale pe baza unor modele teoretice existente sau elaborate de autorii proiectului).

7) Abordarea problemei gestionării bazelor de date prin combinarea tehnicilor teoriei relaţionale cu cele ale inteligenţei artificiale.

3. *De ce un proiect-pilot?*

Cadru de organizare a unor cercetări conduse în planuri care se întrepătrund puternic, proiectul-pilot nu este încă o formă frecvent întîlnită în informatica românească; pe plan mondial și-a dovedit cu prisosință virtuțile.

Vom prezenta, în cele ce urmează, cîteva elemente care au dictat opțiunea.

— Direcțiile de cercetare stabilite sînt departe de a fi independente; condiționarea lor reciprocă este evidentă. Investigarea lor separată, în afara unui cadru care să disciplineze în egală măsură strategiile de abordare, dar și ordonarea rezultatelor în planul sintetic, nu ar fi eficientă.

— Stadiul în care ne aflăm determină succesiuni de faze de cercetare neconforme tiparelor „lineare" obișnuite (documentare, specificații de definire, specificații de realizare, elaborare software). Documentarea este necesară permanent; soluțiile teoretice din literatura de specialitate se pot dovedi neeficiente; există „probleme deschise" recunoscute ca atare, dar pot fi descoperite — în încercările de implementare — noi probleme deschise pentru care eventualele soluții teoretice originale trebuie să fie supuse unor verificări „în laborator".

— În condițiile în care gama de echipamente de calcul crește este normal ca soluțiile software oferite să aibă o portabilitate cît mai ridicată; dar se știe că SGBD-urile fac parte din categoria de produse dependente în mare măsură de caracteristicile memoriilor și ale sistemelor de operare. Ce cale trebuie urmată pentru a minimiza această dependență? Este o problemă ce nu-și poate afla răspunsul în cadrul unei abordări limitate (de exemplu, printr-o temă de cercetare avînd ca obiectiv realizarea unui anumit SGBD, destinat unui anumit calculator).

4. *Cercetări în curs de desfășurare în cadrul proiectului RA*

În anul 1984 a fost demarată ([3]) implementarea unei interferențe relaționale de tip Query-by-Example (QBE), pentru SGBD-ul ARGUS (sistem de tip CODASYL, operațional pe minicalculatoarele românești). Un SGBD pseudo-relațional de tip QBE este în curs de realizare; destinat în principal microcalculatoarelor de 16 biți, soluțiile adoptate urmăresc asigurarea unui grad înalt de portabilitate ([4]).

Alegerea, pentru ambele cercetări menționate, a soluției QBE s-a făcut pe baza unui studiu comparativ ([2]) care a investigat limbajele relaționale QBE, SQL, ALPHA, DEDUCE (din punctul de vedere al expresivității accesibilității riscului de eroare, dificultăților de implementare).

Un prim studiu referitor la mașinile de baze de date este disponibil ([6]). Atingerea obiectivului ambițios, dar absolut necesar, de a defini cerințe pentru o mașină de baze date, este condiționată de implementarea experimentală a unui SGBD relațional; cercetarea nu a fost demarată.

Un studiu referitor la conceperea unor sisteme software care să asigure un grad sporit de „inteligență" SGBD-urilor va formula primele concluzii în decembrie 1985. Apreciem că, în această direcție de cercetare, obiectivul cu cele mai mari șanse de a depăși faza „experiment de laborator" este realizarea unei interfețe în limbaj natural. Investigația este sprijinită substanțial atît de existența sistemului de înțelegere a limbajului natural IURES, cît și de reușita experimentului de cuplare IURES-DATATRIEVE pentru o aplicație concretă ([7]). O cercetare importantă — suport indispensabil pentru toate celelalte intenții — vizează implementarea operatorilor Algebrei Relațiilor ([5]).

Pornim de la premisa că proiectul RA nu poate fi realizat decît printr-o strînsă colaborare cu specialiștii din țară și prin utilizarea tuturor formelor posibile de cooperare internațională.

- Seria AMC se procură din librăriile cu carte tehnică
- Din cursul anului 1984, de la volumul AMC 38, pe coperţi apar cele cîteva domenii atacate, volumele capătînd, astfel, titluri. Această tematizare s-a bucurat de succes nu numai în rîndul informaticienilor ci şi al altor specialişti, ca şi a unor cercuri de studenţi, elevi ş.a. Cicluri continue ca: proiectarea asistată de calculator, analiza sistemelor, calculatoare personale, reţele de calculatoare, note de lectură, service pentru calculatoare, congrese ş.a., realizate de specialişti, atrag noi cititori permanenţi.
- Pentru a vă asigura obţinerea seriei AMC în noile condiţii, recomandăm întreprinderilor să ne trimită pe adresa: EDITURA TEHNICĂ, Piaţa SCÎNTEII 1, comenzi ferme anticipate, semnate de director şi contabil şef, care să colecteze atît volumele necesare documentării unităţii cît şi cererile ferme exprimate de oamenii muncii prin organe sindicale (FUS, CIT ş.a.). Şi cititorii individuali ne pot scrie, indicînd adresa exactă. Noi înaintăm toate aceste comenzi, cu indicaţii de prioritate, centrelor de librării, prin Întreprinderea de difuzare a cărţii (I.D.C.).
- Plata se face la primirea volumelor, nu anticipat.
- În 1984 au apărut volumele 35–44, iar în 1985 (trim. I–II) AMC 45–47.
- În trim. III apar AMC 48–51, iar în trim. IV, AMC 52–54. Preţul unui volum este de 22–25 lei.
- În 1986 sînt prevăzute AMC 55–60.
- Eventualele lămuriri la tel. 18.06.30 sau 17.60.10/2100.


EDITURA TEHNICĂ

- Volum realizat prin cooperare cu Comisia de Cibernetică a Academiei R. S. România, Institutul de tehnică de calcul şi informatică, Institutul de proiectări automatizări, Întreprinderea pentru întreţinerea şi repararea utilajelor de calcul şi cu specialişti din aceste foruri şi unităţi.
- Cu sprijinul Federaţiei Internaţionale de automatizare (IFAC) şi al IIASA–Laxbuberg, şi cu concursul unor specialişti din R.F.G., Japonia, S.U.A., U.R.S.S., R. S. Cehoslovacă, R.P.U., Austria.
- Cu concursul şi al unor specialişti de la Institutul Politehnic Bucureşti, Fabrica de calculatoare.

Lei 26